

Nanofotonische Reservoir Computing met fotonischekristalcaviteiten

Nanophotonic Reservoir Computing Using Photonic Crystal Cavities

Martin Fiers

Promotoren: prof. dr. ir. P. Bienstman, prof. dr. ir. J. Dambre
Proefschrift ingediend tot het behalen van de graad van
Doctor in de Ingenieurswetenschappen: Fotonica

Vakgroep Informatietechnologie
Voorzitter: prof. dr. ir. D. De Zutter
Faculteit Ingenieurswetenschappen en Architectuur
Academiejaar 2012 - 2013



ISBN 978-90-8578-604-7
NUR 965
Wettelijk depot: D/2013/10.500/37

Promotoren:

Prof. Dr. Ir. Peter Bienstman
Prof. Dr. Ir. Joni Dambre

Examencommissie:

Prof. Dr. Ir. Luc Taerwe (voorzitter)	Universiteit Gent, Bouwkundige Constructies
Prof. Dr. Ir. Peter Bienstman (promotor)	Universiteit Gent, INTEC
Prof. Dr. Ir. Joni Dambre (promotor)	Universiteit Gent, ELIS
Prof. Dr. Ir. Wim Bogaerts (secretaris)	Universiteit Gent, INTEC
Prof. Dr. Ir. Dries Vande Ginste	Universiteit Gent, INTEC
Prof. Dr. Ir. Aleksandra Pizurica	Universiteit Gent, TELIN
Dr. Ir. Marc Vanden Bossche	National Instruments Belgium NV/SA
Prof. Dr. Ir. Bjorn Maes	Université De Mons, Faculty of Science

Universiteit Gent
Faculteit Ingenieurswetenschappen en Architectuur

Vakgroep Informatietechnologie
Sint-Pietersnieuwstraat 41, B-9000 Gent, België

Tel.: +32-9-264.33.39

Fax.: +32-9-331.35.93



Dit werk kwam tot stand in het kader van een beurs van
het Bijzonder Onderzoeksfonds (BOF).

Dankwoord

Gent, juni 2013

Martin Fiers

Computers en programmeren hebben me steeds gepassioneerd. Dit was reeds duidelijk toen ik op 14-jarige leeftijd in Q-basic een volledige snake-kloon maakte met map-editor en items. Ook op de universiteit, ondanks het feit dat ik geen computerwetenschappen heb gestudeerd, werd deze interesse duidelijk toen ik m'n thesis (de complexe jacobimethode) mocht starten bij Peter Bienstman. Een belangrijk element van de thesis was namelijk het programmeren van een simulatieprogramma.

Peter Bienstman, die tijdens zijn eigen doctoraat het programma CAMFR heeft geschreven, bleek achteraf er geen groot probleem van te maken als ik tijdens mijn doctoraat opnieuw ging programmeren, met een nieuwe opzet: optische circuits simuleren. Ik had het nodig om m'n onderzoek te doen maar het zou wel even duren om dit te maken. Dit resulteerde in een programma met op dit moment ongeveer 40000 lijnen code. Men kan zich dan afvragen: wat stelt zoveel lijnen nu voor? Wel, ik kan u toch vertellen dat het een werk van lange adem, veel zweet en nachtelijke uurtjes programmeerwerk was.

Thomas, die op dat moment zijn thesis was begonnen in onze groep, zit hier ook wel voor iets tussen. Verdorie toch moest hij componentjes met 4 poorten simuleren en dat ging nu net niet, enkel 2 poortjes lukte. Dit boekhoudkundig probleem heeft ons enkele weken zoet gehouden, maar het resultaat loonde: plotseling konden we optische filters simuleren. Dat hadden we niet in die mate verwacht. Nu wilden we heel grote circuits gaan simuleren, maar daarvoor was het programma te traag. Want ja, we wilden dan 1000 componentjes tegelijk uitrekenen en dan krijg je van die 1000x1000 matrices die je moet inverteren enzovoort. Gelukkig kwam Ken vanuit de reservoir computing groep to the rescue en programmeerde op korte tijd een efficiënte matrix-klasse. En toen hadden we een zeer snelle simulator. De volgende uitdaging was toen we les moesten geven aan een 40-tal studenten en we hadden nog geen grafische

interface om het wat gebruiksvriendelijker te maken. Bijgevolgd heb ik een paar weken nachtelijke uurtjes geklopt om javascript te leren en een eerste concept te maken van de grafische interface (opnieuw bedankt Ken voor de hulp toen). De deadline was krap en ook al hadden we iets min of meer werkend, toch liep het grandioos fout tijdens het practicum. Yannick, bedankt om het op dat moment toch in goede banen te proberen leiden. Ik kan uren blijven doorgaan over de *Caphe*-verhalen, maar daarvoor dient het dankwoord eigenlijk niet. Ik ga hier dan ook effectief beginnen met mensen te bedanken.

Allereerst, heel veel dank aan Peter Bienstman, als promotor, om me te vertrouwen als ik voor de zoveelste keer een zijsprong nam in m'n doctoraat. En ook bedankt dat ik me na m'n doctoraat kan blijven bezighouden met wat ik graag doe, namelijk het product dat we toen gecreëerd hebben verder uitwerken tot een commercieel product! Samen met het programma dat Wim oorspronkelijk begonnen is (*IPKISS*), gaan we een boeiende periode tegemoet. Wim, Pieter, Danae, Joris en Erwin, en de coaches Greg en Bruno: bedankt! Binnenkort lopen we tussen de EDA giganten.

Wat houdt zo een doctoraat dan in naast—in mijn geval— veel programmeren? Voor de lezers die niet uit de groep komen: een doctoraat is typisch het eindresultaat van +/- 4 jaar onderzoek, en neen, 't is niet altijd even gemakkelijk als sommigen van jullie beweren. OK, we moeten geen belasting betalen, en de werkdruk is over het algemeen acceptabel, maar ons brein ziet toch af hoor in die periode! Ik wil die lezers dan ook bedanken voor hun belastingsgeld.. hum, ik bedoel, voor hun vertrouwen in wat we doen en te geloven dat het toch ook de mensheid een beetje vooruit helpt. Meer hierover kan je lezen in dit doctoraat. Veel plezier!

Menig collega heeft gelachen toen Kristof een zoveelste poging deed tijdens de groepsvergaderingen om uit te leggen wat reservoir computing nu betekent. Intussen heb ik het gevoel dat jullie de ontkenningfase voorbij zijn, en dat jullie reservoir computing al iets meer *au serieux* nemen. Aan die collega's die nu reeds wegzijn, Koen, Joris, Wout, ...: bedankt om de nieuwe mensen zo vlot te ontvangen en niet al te hard te lachen met reservoir computing.

Een zeer dikke dankjewel aan Ilse en Kristien van het secretariaat! Wat jullie doen mag niet onderschat worden. Wat ben ik blij dat ik steeds bij jullie terecht kon voor al mijn problemen groot en klein!

Karel en Elewout! Glazen water omgekeerd op elkaars bureau zetten kan toch amusant zijn. Bedankt Karel om samen een huisje te delen, en erna je plaatsje af te staan aan Leen. Je droge humor weergalmt nog steeds door ons huis, maar gelukkig ben ik nu alweer van je af zodat ik geen stapels afwas meer moet verbergen. Bedankt Elewout voor de humoristische toon op het werk en het vele werk dat je stak in de afscheidscadeautjes. Jammer dat ik niet weg was voor jou ;) Diedrik, ooit worden we nog rijk met bitcoins! Hadden we maar niet

geluisterd naar onze collega's en er toen een 10000-tal gekocht.

M'n bureaugenoten zorgden in m'n doctoraatsjaren voor het nodige amusement. Ieder had zo z'n eigen stijl. Yannick, steeds zo enthousiast, sportief en plichtsbewust. Zonder dat je het weet wordt je nog eens CEO. Bart, wandelend rekenmachine, kon ik jou in m'n computer steken, dan had ik Caphe niet meer nodig. Je enorme berg kennis verbaast me nog steeds, alsook je opmerkelijke nieuwsgierigheid. Marie, altijd met een glimlach op de bureau, en voor ons plantjes zorgen met de overschot van de thee. Nebiyu, for always smiling. I hope you are fine in Leuven with your wife and child! Gunay, your turkish massages are the best, unfortunately there's only a few people that agree with me!

Voor de mensen van ELIS (reservoir lab): ik heb me altijd zeer welkom gevoeld door iedereen uit jullie groep. Er was geen enkel moment dat ik niet kon langskomen met lastige machine learning vragen, van het begin van m'n doctoraat tot de dag voor m'n interne verdediging. Ik voel me altijd in een mini silicon valley als ik naar beneden kom (althoewel de groep boven jullie ironisch gezien niks anders doet dan met silicium werken). Bedankt! Ook Joni en Ben, bedankt voor jullie immer kritische blik bij het nalezen van m'n papers en business-plan pogingen, die heb ik ten zeerste geapprecieerd.

Thank you, Caphe-testing colleagues, Bendix, Sarvagya, Alfonso, Sam, Peter DH, Thijs, Imad, Eva (in no particular order) and probably many others which I have forgotten here. Thank you Raphaël, for not testing it (I had to put you in somehow). Thanks Pauline and Kristof, for being such good listeners with all my problems. Thomas, onze wiskundige discussies van topniveau —waaruit vaak dan blijkt dat ik het weer fout heb geïnterpreteerd— blijf ik echt leukvinden!

Ook dank aan Podo, Jack, Dolfie, Boelie, Giry De Stomme Van Assche, Milky, Neel, Nitram enzovoort. Bart, Jeroen, Floris, Sofian, Jonas, Joke, Xavier, jullie zorgen altijd voor een goede sfeer, wherever we go!

Ik bedank ook m'n ouders om er altijd voor me te zijn geweest. Hoeveel keer jullie logistieke problemen hebben opgelost, het is haast een wereldrecord denk ik. Gilles, bro, bedankt voor de 'go kick some ass, brother!' nota, die me tot Belgisch kampioen heeft gekatapulteerd, en Geraldine, zussie, voor je optimisme en de immer-vrolijke noot in huis.

Daarnaast bedank ik ook Noella, Ronnie, Els en Raph die steeds geïnteresseerd naar mijn doctoraatsverhalen geluisterd hebben. Jullie geloofden steeds met volle overtuiging —of dat doen jullie mij toch geloven— dat ik iets nuttigs deed! De Limburgse gastvrijheid heeft een wereld voor me open doen gaan (omgekeerd met m'n onderzoek geldt dat waarschijnlijk niet ;).

Leen. Er is zoveel waar ik je voor wil bedanken. De meeste ken je wel al. Als ik er dan eentje moet uitnemen: bedankt voor je vele geduld bij alles wat ik doe. De manier waarop ik hier nu sta is dankzij jou: gelukkig, zelfverzekerd. De wereld ligt aan ons voeten.

Table of Contents

Dankwoord	i
Nederlandse samenvatting	xxix
1 Machinaal leren	xxix
2 Fotonica	xxx
3 Machinaal leren en fotonica	xxxix
English summary	xxxv
1 Machine learning	xxxv
2 Photonics	xxxvi
3 Machine learning and photonics	xxxvi
1 Introduction	1
1.1 Information processing: the current state	4
1.2 Nanophotonics	6
1.2.1 Fabrication of nanophotonic chips	7
1.2.2 Nonlinearities	8
1.2.3 Building blocks for optical neural networks	9
1.3 Goal	10
1.4 Thesis outline	11
1.5 Publications	11
References	15
2 Reservoir Computing	19
2.1 Information processing	20
2.2 Machine Learning	23
2.2.1 Different learning methods	23
2.2.2 Artificial Neural Networks	24
2.2.2.1 Neuron types	24
2.2.2.2 Network topologies	27
2.2.2.3 Hardware implementations of artificial neural networks	28

2.3	Reservoir Computing	29
2.3.1	Mathematical description of the reservoir	33
2.3.2	Training the reservoir	34
2.3.3	Off-line training	34
2.3.3.1	Regularization	36
2.3.3.2	Cross-validation	37
2.3.3.3	Unbalanced datasets and Fisher relabeling	38
2.3.4	On-line learning	40
2.3.4.1	Generating periodic patterns using FORCE	42
2.4	Software framework	43
	References	43
3	Photonic Crystal Cavities	49
3.1	Photonic crystals: principle	50
3.1.1	Periodicity and band gaps	50
3.1.2	3D vs 2D photonic crystals	52
3.1.3	Nonlinearities	53
3.1.4	Photonic crystal waveguides and cavities	54
3.2	Simulation of photonic crystal cavities	55
3.2.1	Coupled Mode Theory	55
3.2.1.1	Equations	56
3.2.1.2	A single cavity	57
3.2.1.3	Dynamics of two coupled cavities in serie	58
3.2.2	FDTD	61
3.2.2.1	2D photonic crystal with a rectangular lattice	61
3.2.2.2	2D photonic crystal waveguide and cavity	61
3.2.2.3	Behavior of a photonic crystal cavity with Kerr nonlinearity	63
3.2.2.4	2 cavities in a row	65
3.3	Measurements	68
3.3.1	Design and fabrication	68
3.3.2	Measurement and post-processing	69
3.3.3	Results	70
3.3.4	Acknowledgements	70
3.4	Conclusions	71
	References	71
4	Caphe: A framework for simulating large networks of optical components	75
4.1	Numerical modeling	76
4.2	Model	79

4.2.1	Scatter matrices	79
4.2.2	Extension for S-matrices	82
4.2.3	Carrier modulation	83
4.2.3.1	Comparison with circuit envelope simulation . . .	84
4.2.4	Generalized source term	85
4.3	Towards a circuit	85
4.3.1	Generalized connection matrix	85
4.3.2	Integration in time-domain	88
4.4	Optimizations	89
4.4.1	Optimizations in the frequency domain	89
4.4.2	Improving the simulation speed in the time domain	90
4.5	Examples	92
4.5.1	Coupled Resonator Optical Waveguide	92
4.5.2	Dynamics of three coupled ring resonators in a feedback loop	94
4.6	Constructing a nanophotonic reservoir	95
4.6.1	Hardware topology	95
4.6.2	Implementation of the network	98
4.6.2.1	Creating the circuit	98
4.6.2.2	Flattening the network	100
4.7	Alternatives	101
4.8	Conclusion	102
	References	102
5	Isolated spoken digit recognition using photonic crystal cavities	107
5.1	Isolated digit recognition: task description	108
5.1.1	Pre-processing	108
5.1.2	Winner-takes-all	110
5.2	Previous work in nanophotonic reservoir computing using SOAs .	110
5.2.1	Semiconductor Optical Amplifiers	112
5.2.2	Topology	112
5.2.3	Summary of previous results	113
5.2.4	Simulation framework	113
5.3	Comparison between photonic crystal cavities and SOAs	114
5.4	The Jacobian of a system of photonic crystal cavities	117
5.4.1	Deriving the Jacobian for a CMT system	118
5.4.2	Interpretation	119
5.5	Simulation results	120
5.5.1	Default parameters	120
5.5.2	Influence of the cavity lifetime	121
5.5.3	Fixed phase versus random phase	122
5.5.4	Influence of the topology	125

5.5.5	Influence of the cavity type	126
5.5.6	Influence of detuning and input power	129
5.5.7	Influence of fabrication errors	131
5.6	Conclusions	131
	References	133
6	Generating periodic patterns	137
6.1	Task description	139
6.2	Performance measure	140
6.3	From discrete time to continuous time	141
6.4	From real-valued to complex-valued reservoirs	143
6.4.1	Continuous and complex-valued	145
6.5	The full Photonic Crystal Cavity reservoir	147
6.5.1	Phase reflection of the resonator	148
6.5.2	Delays	149
6.5.3	Splitters	150
6.5.4	Restricting the readout to the power only	153
6.5.5	Resonance frequency changes due to fabrication imperfections	154
6.5.6	Network size	155
6.6	Information processing capacity	155
6.7	Hardware challenges	157
6.8	Choosing phenomenological parameters instead of physical parameters	158
6.9	Conclusion	158
	References	160
7	Conclusions and Perspectives	163
7.1	Summary	163
7.2	Perspectives and future work	165
7.2.1	Alternative training methods	165
7.2.2	New benchmark tasks and applications	165
7.2.3	Improved modeling	166
7.2.4	Measurements and experimental setup	166
	References	168
A	Derivation of the efficient computation of the NRMSE	169

List of Figures

1	FDTD simulatie van een fotonischekristalcaviteit.	xxxi
2	Illustratie van een reservoir computer. De invoer (links) wordt aan het reservoir gevoed (midden). Een uitleeslaag (rechts) extraheert informatie van het reservoir.	xxxiii
3	Illustratie van een reservoir met terugkoppeling van de uitvoer. Het principe is hetzelfde als bij een normaal reservoir, maar de uitvoer wordt teruggekoppeld naar de invoer.	xxxiii
1	FDTD simulation of a photonic crystal cavity.	xxxvii
2	Illustration of a reservoir computer. The inputs (left) are fed to the reservoir (middle). A readout layer (right) then extracts information from the reservoir.	xxxviii
3	Illustration of a reservoir with output feedback. In addition to the original reservoir, the output is fed back to the input.	xxxix
1.1	Illustration of the capacity limit of electronic wires. The bandwidth is proportional to A/l^2 , which means that there is a physical limit on how much data can be sent over a certain distance given a limited area A. In modern microprocessors, we are close to this limit.	5
1.2	The electromagnetic spectrum. The most interesting part for photonics is in the visible to near-infrared window. Silicon becomes transparent around 1110 nm, and two wavelengths often used in telecommunication are wavelengths around 1310 nm and around 1550 nm. Other material systems such as Silicon Nitride operate in the visible region.	6
1.3	The layers of a standard SOI stack. The thickness of the bottom Silicon layer depends on the way the SOI stack was fabricated and whether or not the final wafer is thinned. The top layer is patterned to create nanophotonic structures. Typically, etch depths of 70 and 220 nm are used.	7

1.4	Some examples of nanophotonic subcomponents created by optical lithography. These components are building blocks for integrated optical circuits. Because a nanophotonic circuit is planar, crossings (left) are sometimes needed. Tapers (right) are used to spread light from a narrow waveguide to a broad one. On the bottom, Scanning Electron Microscope (SEM) pictures of the fabricated devices are shown.	8
2.1	Image from "The Cat is Out of the Bag: Cortical Simulations with 10^9 Neurons, 10^{13} Synapses". Growth of the Top 500 supercomputers ($N=1$ is the strongest computer, and SUM is the sum of all computing power) overlaid with the results from the paper and a projection for realtime human-scale cortical simulation.	22
2.2	A simple artificial neural network. This kind of network is called a feedforward neural network (with one <i>hidden</i> layer), since all signals propagate in one direction and there are no feedback connections. A neuron can either perform a weighted sum on its inputs and apply a nonlinear transformation (such as a thresholding function or a sigmoid function), or it can be based on differential equations, as in the case of spiking neurons. In the case of a spiking neural network, the connections (in this case also called synapses) can also embed an exponential filter.	25
2.3	Different network topologies for neural networks. (a): a feedforward neural network or multilayer perceptron. (b): a recurrent neural network.	27
2.4	By mapping the original feature space to a higher dimensional feature space, it becomes easier to separate them with a linear plane (hyperplane). In this example, time traces of two (different) spoken digits are shown, which represent the state of the reservoir. Here, it is clear that the constructed linear plane can separate the two digits relatively well (picture courtesy of D. Verstraeten). . . .	31
2.5	A classical discrete-time Reservoir Computing (RC) system. It consists of a Recurrent Neural Network (RNN), called the reservoir, an input layer and a readout layer. The reservoir weights \mathbf{W}_{res} are usually chosen randomly (but globally scaled to reach the desired regime), and are unmodified. Input $\mathbf{u}[k]$ is fed to the reservoir and excites the dynamical system. Features of the dynamical system can be extracted by the linear readout layer. Using a set of known training inputs and desired outputs, the output weights \mathbf{W}_{out} are trained in order to minimize the difference between the actual output and the desired output.	32

2.6	In an unbalanced dataset, the linear separation plane can shift towards the class with more samples. In this illustration two classes, A and B are used. Because there are more samples of class B than there are of class A, the separating plane (dashed lines) will be skewed towards class B. In this case, the example of class B marked in red is classified wrongly. By relabeling the weights (in this case, increasing the strength of label 'A', and decreasing the strength of label 'B'), the resulting separation plane (full line) is more accurate, leading to a correct classification of the red 'B'. This technique is also called Fisher relabeling.	39
2.7	An RNN network with feedback. Note that the output weights $\mathbf{W}_{out}[k]$ are now time dependent. The output is fed back to the input. The purpose of this network is to autonomously generate periodic patterns after \mathbf{W}_{out} has been trained.	43
3.1	A 2D photonic crystal cavity with a line defect (W1 defect).	49
3.2	A 2D photonic crystal with rectangular lattice. The rods have a high dielectric constant, the surrounding has a low dielectric constant. Light propagates in the horizontal direction through a line defect (also called a W1 defect). The lattice is defined by the two vectors \mathbf{a}_1 and \mathbf{a}_2 , both have magnitude a . The rods have a radius defined by $r = 0.25a$ and the defect rods have a radius $r = \frac{0.25}{3}a$. These parameters will be used later on for our simulations.	51
3.3	Image of the y-component of the magnetic field \mathbf{H} for the first TM mode at the X-point. The structure is a 2D rectangular lattice of rods surrounded by a low-index medium, as shown in Figure 3.2(b).	52
3.4	Illustration of the two type of cavities that are used throughout this dissertation. Top: a physical 2D layout of the cavity. Bottom: the schematic representation. Left: an inline coupled cavity. Right: a side-coupled cavity.	54
3.5	Exciting the cavity mode using a point source in an FDTD simulation.	54
3.6	A schematic representation of two coupled cavities in series. The reference planes are chosen symmetrical on both sides of the cavity. The distance from the center of the cavity to the reference plane determines the phase reflection parameter ϕ_k	55
3.7	Steady-state curves of a single cavity with Kerr nonlinearity defined by a characteristic power P_0	58

3.8	Transmission (P_{trans}/P_{in}) and classification of the dual-cavity device for different ϕ . Different regimes are indicated: (S) stable, (BI) bistable, (SP) self pulsing and chaos. Note: in this figure the definition of detuning is chosen opposite, i.e., $\Delta = (\omega - \omega_r) \tau$	59
3.9	Example time-trace for $\phi = 0.2\pi$, $P_{in} = P_0$ and $\Delta = 2$	59
3.10	Time-trace for a series of 3 cavities (shown in inset). $\phi = 0.5\pi$ and $\Delta = 0.75$. Depending on the input power, the dynamics can range from stable to self-pulsing to chaos. This demonstrates the rich dynamics of a nonlinear dynamical system consisting of photonic crystal cavities. In the following chapters, we will make networks of 100s of resonators, and predicting the regions of stability becomes very difficult.	60
3.11	Banddiagram for a 2D photonic crystal with a rectangular lattice (shown in inset (1)) for TM polarization. We traverse the k-space in the 2D plane. One can see that there is a region of frequencies where no light can propagate for any direction (the bandgap). Also, above the light line, no light can propagate. The simulations were performed with MPB [11]. Inset (2) shows the irreducible Brillouin zone for this structure.	62
3.12	Finding the resonance in the photonic crystal cavity using a 2D FDTD simulation. The resonance is of a Lorentzian shape, and by fitting this theoretical curve (equation 3.17) to the normalized transmission we can calculate the linewidth and resonance wavelength of the photonic crystal cavity.	64
3.13	Bistable curve for a nonlinear photonic crystal cavity. The blue dots are the result of the FDTD simulations. This is fitted to the analytical formula for a lorentzian-shaped resonance with nonlinearity (full line in red).	66
3.14	Ez-field of the FDTD simulation of two cavities in a row after 170 optical periods. The system is self-pulsing. Simulation parameters: $P_{in} = 5.6254/P_0$, $d = 14$ (number of rods between the two cavities) and $\Delta = 2.225$	66
3.15	Comparison of the CMT and FDTD simulation. The simplified CMT model is able to accurately describe the physical behavior of two coupled nonlinear resonators.	67
3.16	A 1D wire cavity. The radius increases towards the center according to a parabolic profile.	67
3.17	Measurement of the photonic crystal cavities with $w_{wg} = 0.46\mu m$, $F = 1.1$. Three distinct resonances are found, with a maximum Q-factor of 5067 (the left resonance). See Table 3.4 for more information.	70

4.1	Illustration of several simulation tools when designing a multi-mode interferometer (MMI). An eigenmode solver (top left) calculates the mode profile of a waveguide. This eigenmode is then used as input for a Finite Difference Time Domain (FDTD) simulation (top right). The output of this simulation can be sent to a circuit simulation tool. Also, users might want to perform a part of the simulation using their own code and link these to other tools.	77
4.2	An N-port optical component which is treated as a black box. If the optical component is linear, the input-output relationship is fully determined by the scatter matrix \mathbf{S} .	80
4.3	Structure of a node with N ports. A linear and instantaneous node is described by a scatter matrix \mathbf{S} . State variables (e.g. temperature and free carriers) can be added, accompanied by ordinary differential equations (ODE). In this case the node becomes non-instantaneous and can contain nonlinear behavior.	82
4.4	Illustration of a microring resonator. It consists of two parts: the directional coupler and the (bent) waveguide. We also show the two memory-containing ports, which come from a laser and optical spectrum analyzer (OSA). All memoryless nodes are eliminated from the circuit, so we end up with a small (2x2) generalized connection matrix S of the circuit.	87
4.5	A Coupled Resonator Optical Waveguide (CROW). Each section is subdivided in a directional coupler and two waveguides. Port numbers are shown in the left.	90
4.6	Calculating the frequency response of a passive network. Using KLU, a sparse matrix solver suited for circuit-like matrices, we can easily calculate scatter matrices of very large networks.	91
4.7	Left: topology used to simulate a complex system with ML and MC nodes. Each circle represents a SOA. Splitters are not shown. Right: the simulation time and memory usage increases linearly with the number of SOAs. Clearly there is an advantage by eliminating the ML nodes, both in terms of speed and memory usage.	92
4.8	CROW: Optimizing the κ_i to match a certain filter (left). With process variations, performance deteriorates (right).	93
4.9	Self-pulsation in a single (all-pass) microring resonator.	94
4.10	Dynamics of a system with three (all-pass) microring resonators coupled with a feedback loop (zero roundtrip phase at the signal wavelength), containing two 3dB-splitters, connecting the loop with resp. a source and a detector.	95

-
- 4.11 The proposed topology for the nanophotonic reservoir. Each circle represents a PhCC, and each splitter represents an MMI or Y-junction. Due to hardware restrictions, it is difficult to use a random topology. Especially the fan-in should be minimized, because a large fan-in means a higher sensitivity to process variations and increased design complexity. A regular mesh topology, such as the waterfall topology shown here, minimizes the fan-in and crossings, while keeping a good connectivity. A reservoir based on the waterfall topology has a good performance, can be easily designed and minimizes the amount of fan-in and fan-out. 97
- 4.12 Illustration of the typical splitting ratios in a fully connected node in the waterfall topology (a 50/50 splitter is also called a 3dB splitter). The fan-in and fan-out have been minimized to three. An important design parameter is the splitting ratio S . It is the fraction of power that enters the network from the source, and the fraction of power that goes to the detector. 97
- 4.13 Principle for creating a nanophotonic reservoir using Caphe. The neurons are encapsulated in building blocks (a), which are then connected to other blocks (b), in order to form a reservoir (c). . . . 99
- 5.1 Example time traces for a spoken digit after pre-processing with the Lyon passive ear model. (a) shows the 77-channel output of the Lyon passive ear model for one spoken digit, and (b) shows the same data after multiplying with \mathbf{W}_{in} , and shifting all signals upwards, so they become positive for all timesteps. This is the actual input into the reservoir. In this example, we have normalized the output power such that the maximum power into a node is P_0 . 109
- 5.2 After taking the average over time of the output classifiers, we apply the winner-take-all principle. The winning sample is the one with the highest positive output. In (a), the highest output corresponds to spoken digit '7', which is correctly recognized. In (b), digit '5' is chosen as wrong answer. 111
- 5.3 WER for the isolated digit recognition task, for a network of SOA and a classical hyperbolic tangent network. The SOA network, when simulated in a coherent regime (i.e., using complex-valued signals), performs better than a classical, real-valued (incoherent) hyperbolic tangent network. Clearly, there is an optimal value for the interconnection delay, which turns out to be approximately half of the word length (picture courtesy of K. Vandoorne). 114

-
- 5.4 Input vs output power of an SOA. The input-output characteristic of this SOA resembles the input-output of a hyperbolic tangent function (picture courtesy of K. Vandoorne). 115
- 5.5 Input vs output power of the PhCC cavity. Two detunings are shown: a detuning $\Delta = 2$, for which bistability is observed, and a detuning $\Delta = 0$, which is on resonance. For both cases, an inline coupled (investigated in detail in chapter 3) and side coupled cavity are shown. 115
- 5.6 Step response of an SOA. After a warmup period of 1.5 ns, the source is turned on. The input signal is amplified by the SOA. For this it uses an amount of excited carriers, which decreases the SOA gain. When the source is switched off, the gain recovers to its steady-state value, which depends on the amount of current that is injected in the device. 116
- 5.7 Step response of a photonic crystal cavity at low input powers ($P_{in} = 0.1P_0$). When excited at resonance ($\Delta = 0$), the inline coupled cavity (left) reaches a steady-state value close to unit transmission (it equals unit transmission in the limit $P_{in} = 0$, or when the Kerr nonlinearity is ignored). The side coupled cavity (right) has a steady-state value close to zero at resonance (and again, equals zero in absence of the Kerr nonlinearity). 117
- 5.8 WER as a function of the interconnection delay τ_d , for $\tau = 1.25ps = 12.5\Delta t$ (slow) and $\tau = 0.139ps = 1.39\Delta t$ (fast), and a network with randomized phases. For the cavities with a small lifetime (small τ), the WER shows a clear optimum when the delay is approximately half the duration of a spoken digit. This is in correspondence with previous results using a reservoir of SOAs, which showed the same type of optimum. The optimal WER of 5 percent is comparable to the WER of 4.5 percent that is found for the SOA network. The simulations were performed for a waterfall topology. 121
- 5.9 WER for fixed phases, again in the case for $\tau = 0.139ps = 1.39\Delta t$ (fast, top) and $\tau = 1.25ps = 12.5\Delta t$ (slow, bottom). The thick lines are the result for random phases (the same as in Figure 5.8), the thin lines are the results for different fixed phases (i.e. $0, 0.1\pi, 0.2\pi\dots$). As can be seen from the figure, the influence of the phase on slow resonators is larger, and covers a wider band. For some phases, the performance for small delays is comparable to the optimal performance for a delay of approximately 3 ps. 123

-
- 5.10 The eigenvalue spectrum of the Jacobian for a network of inline coupled resonators, for fixed phases (left), and random phases (right). For fixed phases, depending on the actual phase reflection of the cavities, the spectrum can be completely different. The reservoir performance depends heavily on the actual phase that is used. The WER that are mentioned correspond to the slow cavities, for $\tau_d = 2.5 ps$. The spectrum for the random phases on the right is shown as an illustration. 124
- 5.11 WER as a function of the interconnection delay τ_d , for $\tau = 1.25 ps = 12.5\Delta t$. For the waterfall topology, the rightmost eigenvalues of the Jacobian is closer to the origin than in the case of the same topology with an attenuation of 3 dB, or the swirl topology. As a result, because the eigenvalues are very close to the origin, the reservoir is less responsive to the inputs and the performance is less good for the waterfall topology without attenuation. 125
- 5.12 The eigenvalue spectrum of a system with random phases and $\Delta = 0$. Increasing the attenuation will shift the rightmost eigenvalue towards the left. When the attenuation is very large, all eigenvalues will end up at $(-1/\tau, 0)$. The waterfall with no attenuation has a rather high 'spectral radius'. For the isolated digit recognition task however, this value should not be too high. This explains why the waterfall topology without attenuation, as shown in Figure 5.8, has a lower performance. 127
- 5.13 Word Error Rate (WER) as a function of the attenuation, for side-coupled and inline cavities, for different topologies. The side-coupled cavities perform less than the inline coupled cavities, because their transmission at steady-state is equal to zero (see Figure 5.7). As can be seen from the figure, the topology (swirl vs waterfall) does not influence the performance much, except for low values of the attenuation. The phase was chosen fixed, $\tau = 0.694835 ps$, $\Delta = 0$ and $\tau_d = 0 ps$ 128
- 5.14 WER as a function of the the detuning and input power of the reservoir. For large detunings, the dynamics of the cavity are not so dependent on the power (this is because the resonance shape of the cavity resonance skews towards negative detuning as shown in Figure 5.15). For positive detunings ($\Delta > 1$), the nonlinearity in the system increases with higher powers, which decreases the performance of the reservoir. Note that the very good performance for low power and $\Delta = 2$ is highly dependent on the actual phase that was chosen between the cavities. In this case, it corresponds to the lowest curve of Figure 5.9. 129

-
- 5.15 The transmission as a function of wavelength for different input powers. For higher input powers, bistability is observed. 130
- 5.16 WER as a function of the variation ω_{rand} in the resonance frequency. The variations in ω do not have a significant effect. The interconnection delay $\tau_d=0$ ps. 132
- 5.17 WER as a function of the variation τ_{rand} in the lifetime of the cavity, for different mean cavity lifetimes. When increasing the mean cavity lifetime, we can afford a larger randomness in the cavity lifetime. The interconnection delay $\tau_d=0$ ps. 132
- 6.1 Simplified illustration of the learning sequence. T_1 is the time period of the slowest varying frequency. During warmup, the input of the reservoir is noise, sampled from a uniform distribution. During training, the output weights \mathbf{W}_{out} are adjusted such that the output (black solid line) follows the target signal (gray dashed line). The output weights are unmodified during freerun. The output can have a slightly different frequency than the target. The last samples $y_{test}[k]$ are scrolled over a window of the freerun output, each time calculating the NRMSE. The optimal value of the NRMSE is used as performance for this learning sequence. 140
- 6.2 Normalized Root Mean Square Error (NRMSE) for different τ_0 for the MSO task described in section 6.1. The error bars show the sample standard deviation over 40 simulations ($NRMSE \pm \sigma_{NRMSE}$), and the dominant time constant of the neurons (τ_0) is swept. Top: the performance of the continuous-time reservoir (dotted green) is better than that of the classical reservoir (red). Bottom: Without delay between the neurons or in the feedback loop, the reservoir can respond faster to changes in the output, leading to a better performance. 144
- 6.3 The NRMSE for three reservoirs as a function of the leak rate: standard leaky hyperbolic tangent reservoir with 200 neurons (red, baseline), the same reservoir with complex-valued states (green, dashed) and a standard reservoir with 400 neurons. Clearly, the complex-valued reservoir performs better than the standard reservoir. For most leak rates, the performance is similar to the system with 400 neurons. 145

6.4	Comparison of discrete time and continuous time reservoirs, both for with real-valued and for complex-valued neurons. The performance gain when using both using complex-valued states and a continuous-time reservoir is not significant compared to complex-valued neurons in a discrete-time reservoir or real-valued neurons in a continuous-time reservoir.	146
6.5	The error (NRMSE) after training an optical network of photonic crystal cavities for the MSO task. The phase between the resonators is described by $\phi_j = 0.2\pi + \phi_r \epsilon$, $\epsilon \sim \mathcal{N}(0, 1)$. This is done for different fractions of cavities receiving bias. The more cavities that receive bias, the more the reservoir dynamics are disturbed by strong interactions between the resonators (e.g. self-pulsation), which causes the network to be unable to generate the signal autonomously. Increasing the randomness in the phase reduces the amount of self-pulsation.	149
6.6	Dynamics for a sequence of two coupled resonators (shown in the inset). With increasing delay, the dynamics change from self-pulsation to chaos. A delay of 0.1 ps corresponds to approx. 10 μm on-chip.	150
6.7	Error (NRMSE) of the photonic reservoir for the MSO task. As shown in Figure 6.5, training fails in certain conditions when the phases are fixed and equal to 0.2π . By increasing the delay (100 ps delay is approximately 12.5 μm on-chip), the self-pulsation in a series of two cavities is lost (see Figure 6.6). The conclusion that the training works better when self-pulsing is not present is also valid here.	151
6.8	Illustration of the splitting ratio S . Additional splitters are needed in order to send the signal from the source into the photonic crystal cavity, and from the cavity to the detector.	151
6.9	Dynamics of a series of two coupled resonators for increasing attenuation of the waveguide (αWG) between the cavities. αWG is the one-way power attenuation. For $\alpha WG = 0.2$, the self-pulsation is lost.	152
6.10	Influence of the splitter ratio S (see Figure 4.12 and Figure 6.8). By increasing the splitting ratio, the interaction between two neighboring cavities is decreased. This is advantageous for learning, because the strong self-pulsation which disrupts training disappear (see also Figure 6.9). Parameters: phases random, $P_{bias}=1.3P_0/\sqrt{S}$, $FB = 1.0/\sqrt{S}$	153

6.11 Comparison of reading out the the real and imaginary part (as we have done in previous experiments), or reading out the power of the reservoir.	154
6.12 Influence of the network size on the performance. We have simulated two variations on the mesh topology, once with a square topology and once with a rectangular topology. Clearly the influence of this slight topology change is negligible. It also shows that, for this specific task, 70 resonators are sufficient, and there is no performance gain by using more resonators.	156
6.13 Total information processing capacity of a 5x5 photonic crystal cavity network. The region with low phase randomness and high input powers correspond with self-pulsation regions. These regions are better avoided in order to increase the total capacity, and hence the performance of the system. This conclusion is in line with previous experiments.	157

List of Tables

3.1	Values used for the time-domain FDTD simulations.	62
3.2	Values used for designing the photonic crystal wire cavity.	69
3.3	Parameters used for the fabrication of the 1D wire cavity.	69
3.4	Measurement results for the 1D wire cavity for $w_{wg} = 460\mu m$. . .	71
4.1	List of possible splitters with N ports, where the power is equally distributed over N-1 output ports. For some N, it is not possible to create a lossless splitter. Advanced magneto-optic materials can be used in order to break reciprocity on the SOI platform [35], but make the fabrication more complex.	96
5.1	Default values used in the photonic crystal cavity (PhCC) reservoir for the isolated digit recognition task.	120
6.1	Default values used in the leaky hyperbolic tangent reservoir. . . .	141
6.2	Default values used in the photonic crystal cavity reservoir.	147
6.3	Summary of the Normalized Root Mean Square Errors (NRMSE) calculated in this chapter for the different architectures. DT = discrete time, CT = continuous time. Tanh: classical hyperbolic tangent neurons, PhCC: photonic crystal cavity.	159

List of Acronyms

A

AGC	Adaptive Gain Controllers
AI	Artificial Intelligence
AWG	Arrayed Waveguide Grating

B

BER	Bit Error Rate
BPDC	BackPropagation DeCorrelation

C

CAPHE	CAvity PHEnomenological modeling framework
CMOS	Complementary Metal Oxide Semiconductor
CMT	Coupled Mode Theory
CPU	Central Processing Unit

D

DBR	Distributed Bragg Gratings
-----	----------------------------

E

ESN Echo State Network

F

FDTD Finite Difference Time Domain
FFNN Feedforward Neural Network
FLOP Floating Point Operations
FLOPS Floating Point Operations per Second
FORCE First-Order Reduced and Corrected Error
FSR Free Spectral Range
FWHM Full Width at Half Minimum

H

HWR Half-Wave Rectifiers

I

IL Insertion Loss

L

LSM Liquid State Machine

M

ML Machine Learning

MLP	Multi-Layer Perceptrons
MMI	Multimode Interferometer
MSE	Mean Square Error
MSO	Multiple Superimposed Oscillator

N

NARMA	Nonlinear Autoregressive Moving Average
NN	Neural Network
NRMSE	Normalized Root Mean Square Error

O

ODE	Ordinary Differential Equation
OGER	OrGanic Environment for Reservoir computing
OSA	Optical Spectrum Analyzer

R

RC	Reservoir Computing
RLS	Recursive Least Squares
RNN	Recurrent Neural Network

S

SOA	Semiconductor Optical Amplifier
SOI	Silicon On Insulator
SNN	Spiking Neural Networks

T

TE	Transverse-electric
TIR	Total Internal Reflection
TM	Transverse-magnetic

W

WER	Word Error Rate
-----	-----------------

Nederlandse samenvatting

–Summary in Dutch–

Elektronische toestellen zijn sterk verwoven in onze maatschappij. De meeste toestellen voeren berekeningen uit, zoals het weergeven van een landschap in een computerspel op een computer desktop, het uitvoeren van wetenschappelijke berekeningen op een supercomputer (bijvoorbeeld proteïne-vouwen), of een tekstberichtje tweeten op je smartphone. Al deze berekeningen worden uitgevoerd op een hardware architectuur die is uitgevonden rond het jaar 1936 door Alan Turing: de Universele Turing machine. Samengevat betekent dit het volgende: een centrale processor (Central Processing Unit, CPU) communiceert met geheugen, die zowel de data als het uit te voeren programma bevat, en in- en uitvoer laat het systeem toe om met de buitenwereld te communiceren. In de hieropvolgende jaren is de technologie steeds verbeterd, met snellere en kleinere systemen tot gevolg. De observatie dat het aantal transistoren op een chip iedere twee jaar verdubbeld, heet ook de wet van Moore. Deze door Gordon Moore voorgestelde observatie is tot op heden nog geldig, alhoewel de exponentiële groei stilaan zal beginnen afvlakken.

1 Machinaal leren

Niettemin staande deze computers reeds zeer krachtig zijn, zijn ze niet goed in menselijke taken zoals patronen herkennen in grote datasets, het herkennen van spraak, de motoren van een wandelende robot aansturen enzovoort. Het heeft veel moeite gekost om een programma te ontwikkelen dat, met gebruik van een Turing machine, getraind kon worden om een schaakspel te spelen en om een menselijke tegenspeler te verslaan. Het onderzoeksveld van machinaal leren houdt zich bezig met systemen te bouwen die, net zoals de mens, kunnen generaliseren en leren van voorbeelden. Een veelgebruikt hulpmiddel in dit onderzoeksveld is een artificieel neurale netwerk. Dit is een systeem dat bestaat uit vele neuronen (typisch 1000 of meer), die via synapsen verbonden zijn met elkaar. Deze neuronen zijn vaak niet-lineair, en het resulterende niet-

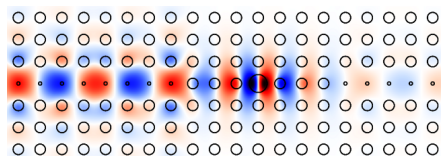
lineaire systeem kan vaak nuttige berekeningen verrichten. Vele van de vooraf vermelde problemen kunnen met deze systemen opgelost worden als het correct getraind wordt. Met een techniek die Reservoir Computing heet, wordt het heel eenvoudig om deze systemen te trainen.

Meestal worden deze systemen op een computer gesimuleerd, waardoor ze niet vermogenefficiënt zijn. Daarom gebeurt er onderzoek naar zogenaamde neuromorfische componenten: dit zijn toestellen die bestaan uit kleine bouwblockjes die geïnspireerd zijn door het menselijk brein. In tegenstelling tot de Turing machine werkt dit toestel asynchroon, waardoor geen kloksignaal moet gedistribueerd worden op de computerchip. Bijgevolg wordt heel veel vermogen bespaard. Als tweede voordeel geldt dat deze chips informatie inherent op een parallelle manier verwerken, in tegenstelling tot de sequentiële werking van een Turing machine. Dit betekent dat de informatieverwerking potentieel sneller kan verlopen.

2 Fotonica

Tegenwoordig is het transferreren van informatie tussen of binnen computerchips verantwoordelijk voor meer dan 50% van de vermogenconsumptie van de totale chip. Het optimaliseren van het vermogenbudget is dus zeer belangrijk. Deze interconnecties vormen de grootste beperking voor electronica, omdat de bandbreedte uiteindelijk wordt gelimiteerd door fysische processen van de halfgeleidermaterialen. Voor een gegeven chipoppervlakte is steeds een maximale bandbreedte, en dus een limiet op de informatieoverdracht naar de chip, en op de chip zelf.

Fotonica biedt een oplossing op dit probleem. Gezien de draagfrequenties van de optische signalen enkele grootteordes hoger liggen, maar toch transparant zijn voor de materialen die typisch gebruikt worden in de halfgeleiderindustrie, zijn de bandbreedtes ook enkele grootteordes groter. De informatie wordt dan gemoduleerd op deze zeer hoge draagfrequenties. Op deze manier kan informatie via licht, zonder veel verlies, getransfereerd worden over verschillende honderden kilometers in een optische vezel. Fotonica is dus veel efficiënter in het transferreren van informatie over lange afstanden, en daarom wordt het dus ook gebruikt in de ruggengraat (backbone) van het internet, een stelsel van zeer snelle computerverbindingen waarlangs het grootste deel van het gegevensverkeer verloopt. Geleidelijk aan komen er producten op de markt die het optisch signaal tot bij de huisdeur brengen. Dit heet Fiber To The Home (FTTH). Het onderzoek naar on-chip interconnecties krijgt tegenwoordig zeer veel aandacht, omdat dit potentieel het verbruik van de chip kan verminderen, en de signaalbandbreedtes verhogen.



Figuur 1: FDTD simulatie van een fotonischekristalcaviteit.

3 Machinaal leren en fotonica

In deze doctoraatsthesis combineren we de zeer hoge bandbreedtes van fotonica met reservoir computing. We bestuderen een neuromorfische component, gebaseerd op fotonischekristalcaviteiten, een optisch bouwblok die in de onderzoekswereld van de fotonica regelmatig wordt gebruikt. Een dergelijke bouwblockje slaat optische energie op in de mode(s) van de caviteit. Afhankelijk van de kwaliteitsfactor (Q-factor) van de resonantie kan licht voor een lange of korte tijd opgeslaan worden. We spreken over tijdschalen van 1-40 picoseconden. Doordat hoge energieën worden opgeslaan, kunnen niet-lineaire effecten optreden. Het neurale netwerk dat we construeren is dus een niet-lineair dynamisch systeem, net zoals de andere artificiële neurale netwerken die in de literatuur worden bestudeerd. Het type niet-lineariteit is dus wel verschillend van de klassieke neurale netwerken. In dit doctoraat concentreren we ons op de Kerr niet-lineariteit, een zeer snelle niet-lineariteit (typisch enkele femtoseconden) die de brekingsindex van het materiaal lokaal aanpast, proportioneel met de lokale intensiteit van het optisch veld. Samengevat bestaat dit doctoraat uit vier bijdragen.

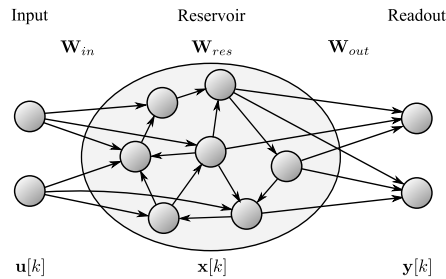
Fotonische kristal modellering In hoofdstuk 3 simuleren we deze fotonischekristalcaviteiten met behulp van twee simulatiemethodes. De eerste is de zeer accurate, maar wel computationeel intensieve eindige differentie tijdsdomein (Finite Difference Time Domain, FDTD) methode (zie Figuur 1). De tweede methode is de benaderde gekoppelde mode theorie (Coupled Mode Theory, CMT). We tonen aan dat we de Kerr niet-lineariteit kunnen reproduceren met het benaderde model (met een gedrag dat zeer goed overeenstemt met de FDTD simulaties). We bekijken ook de dynamica van twee in serie gekoppelde caviteiten. Door het niet-lineaire Kerr effect zal dit systeem in sommige omstandigheden zelf-pulseren, een gedrag dat we met beide methodes kunnen reproduceren. We concluderen dat we de fotonischekristalcaviteiten met goede nauwkeurigheid kunnen simuleren met de benaderde gekoppelde mode theorie, en we zullen deze simulatiemethode ook intensief gebruiken doorheen de komende hoofdstukken.

Nanofotonische modellering Om een groot nanofotonisch reservoir te kunnen simuleren, is een simulatieraamwerk nodig om dit efficiënt te doen. Daarom is een nieuw raamwerk ontwikkeld dat zeer efficiënt niet-lineaire optische circuits kan simuleren, zowel in het tijds- als in het frequentiedomein (hoofdstuk 4). Het resulterende programma, *Caphe*, is niet alleen nuttig voor reservoir computing, maar in veel applicaties binnen de nanofotonica, zoals het ontwerpen van optische filters, het onderzoeken van CMT-gebaseerde modellen en voor applicaties in de telecommunicatie, waarbij lasers, modulators en detectoren gecombineerd worden tot een systeem.

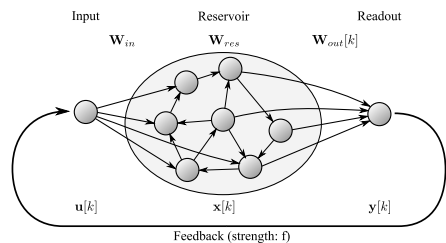
Taak 1: spraakherkenning Met behulp van ons nieuw simulatieraamwerk, simuleren we een groot reservoir bestaande uit fotonischekristalcaviteiten. Een illustratie van het systeem wordt weergegeven in Figuur 2, waar iedere cirkel een neuron voorstelt. Als eerste taak bespreken we de geïsoleerde gesproken cijfers taak, een standaard referentietask die vaak wordt besproken in de reservoir computing literatuur. Onze experimenten¹ zijn gebaseerd op eerdere experimenten van K. Vandoorne, die een nanofotonisch reservoir van optische halfgeleider versterkers (Semiconductor Optical Amplifiers, SOAs) heeft gesimuleerd (tot zover onze kennis reikt was dit de eerste keer dat een nanofotonisch reservoir werd voorgesteld in de literatuur). We concluderen dat er een optimale interconnectievertraging optreedt, waarbij het foutpercentage in voorspelde cijfers 4.5% is. Dit is gelijkaardig aan de resultaten van het SOA netwerk, en beter dan de beste klassieke hyperbolische tangens reservoirs. Uit ons onderzoek blijkt ook dat de fotonischekristalreservoirs het beste werken als ze dicht bij het lineair regime werken, dus als de ingangsvermogens laag zijn.

Taak 2: signaalgeneratie taak Met hetzelfde soort reservoir genereren we nu periodische signalen. Dit doen we door het reservoir te trainen met een nieuwe leertechniek. De opstelling is fundamenteel anders, omdat we de uitvoer van het reservoir terugvoeden naar de invoer, zoals geïllustreerd in Figuur 3. Indien de uitvoergewichten (W_{out}) goed worden getraind kan het systeem zelfstandig arbitraire periodische signalen genereren, zonder dat de gewichten verder moeten worden aangepast na de training. De vernieuwing zit in het feit dat we een fotonischekristalcaviteit reservoir gebruiken, in plaats van de klassieke discrete tijd hyperbolische tangens reservoirs. We concluderen dat de nieuwe leertechniek ook van toepassing is op ons fysisch nanofotonisch reservoir, en dat de performantie beter is (voor evenveel neuronen) dan die van een

¹In de wereld van machinaal leren spreekt men vaak van experimenten indien men iets simuleert. In de fotonicawereld spreekt men van een experiment als er een gefabriceerde chip wordt uitgemeten. In dit doctoraat gebruiken we voornamelijk de machinaal leren conventie. De resultaten zijn dus voornamelijk afkomstig van simulaties.



Figuur 2: Illustratie van een reservoir computer. De invoer (links) wordt aan het reservoir gevoed (midden). Een uitleeslaag (rechts) extraheert informatie van het reservoir.



Figuur 3: Illustratie van een reservoir met terugkoppeling van de uitvoer. Het principe is hetzelfde als bij een normaal reservoir, maar de uitvoer wordt teruggekoppeld naar de invoer.

klassiek reservoir.

De experimenten die in dit doctoraat zijn uitgevoerd tonen theoretisch dat we een neuromorfisch toestel kunnen maken gebaseerd op fotonische kristal-caviteiten. Dit kan leiden tot een nieuwe reeks neuromorfische toestellen die sneller en vermogen efficiënter zijn dan de software-gebaseerde alternatieven. Deze kunnen gebruikt worden om complexe taken op te lossen zoals spraakherkenning, het leren van arbitraire periodische signalen enzovoort.

English summary

Electronic devices are everywhere in our lives. Most of these devices perform some sort of computation, for example rendering a landscape in a video game on a desktop computer, performing scientific calculations such as protein folding on a supercluster, or tweeting a text message on your smartphone. Almost all of them rely on a hardware architecture that was invented around 1936, called the Universal Turing machine, invented by Alan Turing. Loosely speaking, a central processing unit (CPU) communicates with memory, which contains both the data and the program to be executed, and input/output allows the system to communicate to the outside world. In the years that followed, each improvement has focused on miniaturizing these systems, making them faster and processing more data. This scaling obeys Moore's law, stated by Gordon Moore, which says that the number of transistors on a computer chip would double roughly each two years. Even today, this observation is still valid, although the growth will eventually slow down.

1 Machine learning

These computers, although already extremely powerful, are not very good at human-like tasks: finding patterns in a huge amount of data, recognizing speech, controlling the actuators of a walking robot, and so on. It took a great amount of effort to create a program, using the principles of the Turing machine, that could play a game of chess and defeat a human player. The field of machine learning is a research field that tries to build systems that are able to generalize, and learn from examples, much like humans can. A tool that is commonly used in this field is an artificial neural network. This is a system that consists of a large number of neurons (typically 1000 or more), which are connected to each other through synapses. The neurons are usually nonlinear. The resulting nonlinear dynamical system is able to perform computation, and it appears, when properly trained, to be able to cope very well with several of the aforementioned problems. A subfield of this research is called reservoir computing, which makes the training of these systems particularly easy.

Because these systems are typically simulated on a computer, they are not really power efficient. For this reason, researchers are trying to create so-called neuromorphic devices, i.e., chips that contain building blocks that are directly inspired by neurons, and that do not follow the general pattern of the Turing machine. Because such a system operates in an asynchronous manner, it can avoid the energy consumption due to clock distribution. Also, these chips are inherently parallel, similar to the artificial neural network, as opposed to the sequential operation of a Turing inspired machine.

2 Photonics

Nowadays, optimizing the power budget is very important. Data connections, i.e., transferring data on or between chips, take up more than 50% of the total power consumption of a computer chip. These interconnects pose a huge bottleneck for electronics: the electronic bandwidth is ultimately limited by physical processes of the semiconductor material, and for a given square inch of die, there's only so much information that can get on or off the chip.

Photonics offers a promising solution to the interconnect problem. It does not have the bandwidth limitations that electronics have, because the carrier frequencies are a few orders of magnitude larger, yet these optical signals are fully transparent for these high frequencies. This means that light can transfer a data density that simply cannot be reached on the same wire using electronics. With photonics, one can also transmit light over 100s of kilometers of fiber without much loss, so photonics is much more power efficient for transferring data over long distances than electronics. Photonics already takes care of the internet backbone, and it is getting used more and more in end products such as Fiber To The Home (FTTH). Also, the research in on-chip optical interconnects is gaining much attention, again because it can improve the bandwidth and reduce the power consumption.

3 Machine learning and photonics

In this dissertation, we combine the extremely high bandwidths of photonics with reservoir computing. We study a hardware neural network based on a specific type of optical component called the photonic crystal cavity. This component stores energy in the mode(s) of the cavity. Depending on the quality factor (Q-factor) of the resonance, light can be stored for a short/long time, on the order of 1-40 picoseconds. Because cavities can store high amounts of energy, it becomes possible to study nonlinear effects that only occur for high powers.

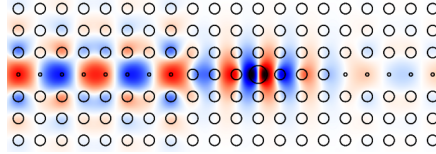


Figure 1: FDTD simulation of a photonic crystal cavity.

The neural network constructed from these cavities is therefore a nonlinear dynamical system, although the nonlinearities are different from the ones we encounter in classical artificial neural networks. In this dissertation, we focus on the ultra-fast (on the order of femtoseconds) Kerr nonlinearity, which changes the material refractive index locally proportional to the intensity of the field. Summarized, this dissertation exists of four contributions.

Photonic crystal modeling In chapter 3, we simulate these photonic crystal cavities using two methodologies: the accurate, but computationally intensive Finite Difference Time Domain (FDTD) (see Figure 1), and the approximated Coupled Mode Theory (CMT). We show that we can reproduce the Kerr nonlinearity with the approximated model (with a quasi-perfect behavior), and we investigate the dynamics in a series of two coupled cavities. The nonlinear dynamics cause the system to self-pulsate for certain parameters, an effect that we can reproduce using both methodologies. The conclusion is that, with good accuracy, we can model these resonators using the approximated coupled mode theory, which we will use extensively throughout the next chapters.

Nanophotonic modeling As a second step towards simulating a large nanophotonic reservoir, we created a framework for the efficient simulation of (optionally highly nonlinear) optical circuits, both in the time and in the frequency domain (chapter 4). The resulting framework, *Caphe*, is not only used for reservoir computing, but for many applications in the field of nanophotonics, mainly for designing optical filters, investigating CMT-based models and for telecommunication applications (laser-modulation-detection).

Task 1: speech recognition Using our novel framework, we simulate a nanophotonic reservoir with photonic crystal cavities (a reservoir typically looks like Figure 2, where each small circle represents a neuron). We first perform the isolated spoken digit recognition task, a commonly used benchmark in reser-

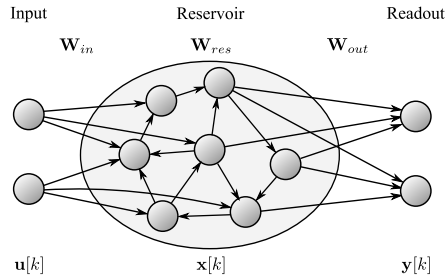


Figure 2: Illustration of a reservoir computer. The inputs (left) are fed to the reservoir (middle). A readout layer (right) then extracts information from the reservoir.

voir computing. Our experiments² were guided by previous experiments of K. Vandoorne, who used a nanophotonic reservoir of Semiconductor Optical Amplifiers (SOAs) to solve the same task (which was, to the best of our knowledge, the first time a nanophotonic reservoir was proposed). We conclude that there is an optimal interconnection delay, which produces Word Error Rates (WER) of about 4.5%, which is similar to the SOA network, and better than state-of-the-art classical hyperbolic tangent reservoirs. Also, we find that the photonic crystal cavities work best close to the linear regime, i.e., when the input powers are not too high.

Task 2: signal generation task Using the same reservoir, we generate periodic signals by training the reservoir with a novel learning method. This setup is fundamentally different from the previous one, because we now feed back the output of the reservoir to the input, as shown in Figure 3. By properly training the weights of the readout layer of the reservoir (W_{out}), the system can autonomously generate arbitrary periodic signals after training without further modifications to the reservoir. The novelty is that we have used our photonic crystal cavity reservoir, instead of the typically used hyperbolic tangent discrete-time reservoirs. We conclude that the new learning method is also applicable for this physical nanophotonic reservoir, and that the performance, for the same number of neurons, is better than the performance of the classical hyperbolic tangent reservoirs.

The experiments that were conducted theoretically show that we can create a neuromorphic device based on photonic crystal cavities. This can lead to a

²In the machine learning literature one typically uses the term experiments, when simulations are performed. In the photonics literature, an experiment typically means actually measuring a physical device. In this dissertation we mainly use the machine learning convention. This means most results in this dissertation are the result of simulations.

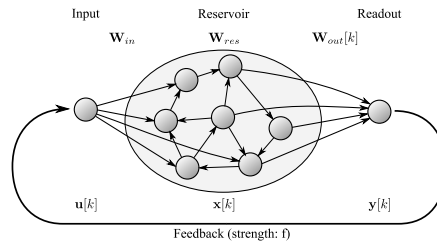


Figure 3: Illustration of a reservoir with output feedback. In addition to the original reservoir, the output is fed back to the input.

new breed of neuromorphic devices that are more power-efficient and faster than software-based equivalents in solving difficult tasks such as recognizing speech, learning arbitrary periodic patterns and so on.

1

Introduction

Over the past decades the amount of research related to the human brain has grown tremendously. Why are people interested in this type of research? The answer is quite simple: humans are really good at ... human-like tasks: steering a car, distinguishing a cat from a dog, (un)successfully reading the hand-writing of a colleague and so on. All these tasks are very difficult to solve with a regular computer which uses a pre-determined step-by-step algorithm. Let's take the example of an image recognition task: suppose we want to create an algorithm that will distinguish the picture of a cat from the picture of a dog. You could write down properties of a cat, being generally more furry, smaller, and with a longer tail, and then try to detect these features in the picture. On that basis, the computer then decides whether the picture is showing a cat or a dog.

But imagine that suddenly you add an additional *class*, let's say an elephant: then again you'll be looking for unique properties (for example the big ears) and add them to your algorithm. And so on. All of this is very cumbersome, and it is exactly these type of questions that sparked the research of *artificial intelligence* and *machine learning*. With Machine Learning (ML), you can feed in thousands of images of different animals (optionally together with the correct classification), and let the computer itself become able to learn the features that distinguish one animal type from the other. If the training has succeeded, it will be able to associate unseen images with the correct animal. All of this can happen without writing an explicit algorithm that is dedicated to this task.

Machine learning has been used successfully to perform a wide variety of tasks, ranging from speech recognition and hand writing recognition to robot locomotion, epileptic seizure detection and so on. Research in this field has looked for inspiration in different areas. For instance, mathematically inspired methods such as kernel machines [1] were fueled by the research field of statistics. Bayesian Networks (BN), based on probabilistic graphical models, are often used to solve decision problems under uncertainty (for example: given a set of symptoms, what is the disease), and Dynamic Bayesian Networks (DBN) are used for temporal problems.

Artificial Neural Networks (ANN) are a tool for information processing, and use more biologically plausible models inspired by neuroscience. The latter is now an extensive research area with many variations and flavors, one of which is Reservoir Computing¹.

Despite these achievements, the capacity of the artificial neural networks we built so far are inferior to the capacity of the human brain. The human brain of a grown-up person has about 10^{11} (one hundred billion) neurons, and each neuron is connected on average to 7000 other neurons through synapses [2]. Still, it only consumes about 23 watt in rest state! Compare this to a modern supercomputer: if we could, with one computer operation, model one connection of the brain, and we would need to model 10^{15} connections, then this would correspond to a computational power of one petaflop². Although it is very difficult to speak of an average firing rate [3], let's assume for simplicity that a neuron fires 100 times per second. This means we need a supercomputer with 100 petaflops to model a human brain. To see things in perspective: the fastest supercomputer available at the time of writing (see [4]) is the Sequoia, at Livermore, and delivers an impressive 16.3 petaflops for a power consumption of 7890 kilowatt. Nature is far ahead of us here.

Of course, this has to do with the standard architecture of a modern computer. A computer is good at processing large amounts of data, i.e., number crunching, following an explicit algorithm which you tell it to execute. It does not try to replicate the structure of the brain. So you'd need to tell the computer to simulate neurons and synapses using connection matrices, complex models of neurons and so on, which poses a considerable overhead to the computer. If, on the other hand, we completely abandon the normal CPU architecture, and make a hardware architecture that resembles the brain (inspired by neural networks), then we can drastically reduce the power needed to perform these calculations. This is because, in this case, we embed the connection matrices

¹But also a Bayesian Network could be implemented as an Artificial Neural Network.

²A flop is the abbreviation for floating point operations. It is used commonly in computer systems to denote the computational power of a system. One petaflop means 10^{15} floating point operations. More common is the term flops, which refers to floating point operations per second.

and neuron models in the hardware itself. This is exactly the topic of this dissertation: *designing a hardware implementation of a neural network*.

As hardware platform we will use *nanophotonics*. The reason we choose this platform is because it has several advantages over electronic implementations. First, light has an amplitude and a phase, which means that more degrees of freedom are present in the network. This is beneficial for the computational power of the system. Second, in electronics, the signal speed is ultimately limited by capacitors and resistors in the circuit, hence this limits the speed of processing. The available bandwidth in photonics on the other hand is several orders of magnitude larger. Furthermore, photonic circuits have nonlinearities which can operate at the ps or even fs time scale.

It is still too early to tell which will be the killer application for photonic reservoir computing. Until now, most of the research in this field has focused on solving benchmark tasks which are well-known for classical reservoir computing in order to be able to compare the advantages and disadvantages w.r.t. the classical case (i.e., software only). One of these tasks is the speech recognition task [5–8], and it has been shown theoretically that these optical implementations can outperform classical reservoir computing. The research in this dissertation, together with previous research, confirms that optical reservoirs can solve these problems faster and more efficiently.

However, the design tolerances for photonics are very stringent, and have to be taken into account when designing a nanophotonic reservoir. In particular, the refractive index is sensitive to slight variations in thickness of the processed wafers and the actual thickness of the guiding structures. Especially in resonant structures, this can lead to a significant shift of the resonance. If a reservoir is designed to work on a certain resonance, then it is important to make the designs tolerant to these variations.

Furthermore, the focus of previous and current research in photonic reservoir computing has been on off-line learning rules. Nevertheless, another important class of learning rules, the on-line learning rules, provide a way to immediately feed back information about the dynamics of the system during training. Accounting for these dynamical effects during training can be very beneficial for certain tasks and has not yet been investigated so far in optical reservoir computing. A research paper by D. Sussillo [9] has sparked the interest of the reservoir computing community. In this paper, a new on-line learning rule was proposed that is exceptionally robust for highly dynamical systems—such as our nanophotonic neural network—and that can be used for several applications such as signal generation and an N-bit memory.

The remainder of this chapter is structured as follows: first, we will explain how the current generation of computer chips is reaching its limits after a long history of scaling to smaller dimensions. We show how nanophotonics has the

potential to revolutionize the industry of information processing, because it can overcome these limitations. We then briefly introduce the reader to this field, starting from photonics in general and then moving on towards photonics on a chip, i.e., nanophotonics. After that, we introduce a list of nanophotonic components that could be used for creating a nanophotonic neural network. We then describe the goals of this dissertation, and introduce the thesis outline.

1.1 Information processing: the current state

The data that we have to our disposal nowadays is quasi-unlimited. Mainly by the advent of the internet³ and the World Wide Web (invented by sir Tim Berners-Lee and Robert Cailliau), the amount of data has grown exponentially. Nowadays we look at YouTube videos, stream a movie to our tablet, have video conferences and use video surveillance. Furthermore, computers perform a variety of resource intensive tasks such as image recognition, speech recognition and analyzing the behavior of visitors on a web-site.

The reason why it is possible to keep analyzing the increasing amount of data, is because the semiconductor industry has continuously scaled down the dimensions of transistors, the basic building block of virtually any computational device. A modern CPU has more than one billion (10^9) transistors. For smaller dimensions, the threshold voltage needed to switch the transistors is lower (along with reduced resistance and capacitance), such that they are faster and consume less power. However, the race towards faster processors has actually slowed down (and even halted). This is because it becomes increasingly difficult to fabricate smaller devices. One can increase the speed of a chip by increasing the current, but at some point, chips generate just as much heat as the package is able to dissipate. Also, signal timing becomes very important and for higher speeds it becomes much more difficult to keep correct operation of the different functional blocks on the chip. Moreover, electronics runs into bandwidth limitations: ultimately, there is a limit to the amount of information one can carry over an electronic wire. This is illustrated in Figure 1.1 where we show two electronic wires that transfer information. In this figure, A refers to the total area of all cross-sections⁴. In [10] it is shown that the maximum bandwidth is proportional to the ratio of this surface cross-section of the wires divided by their squared length, i.e., $B \leq PA/l^2$. P is typically around 10^{16} for a resistive-capacitive on-chip wire. The ratio A/l^2 is a fundamental upper bound and is determined by the used materials and system geometry.

³Internet used to be a shorthand for internetworking, which was the result of interconnecting different networks in order to exchange data.

⁴We could, instead of using one large cable of cross-sectional area A , use several small cables of the same total cross-sectional area A , and obtain the same total bit-rate capacity B .

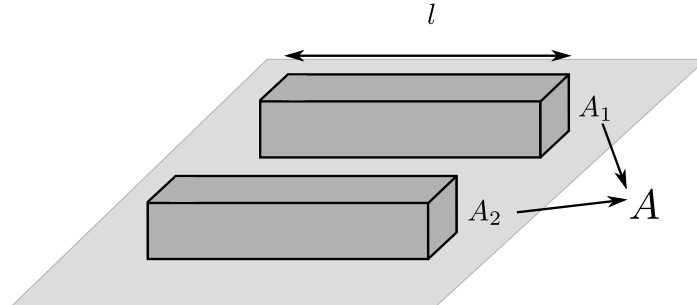


Figure 1.1: Illustration of the capacity limit of electronic wires. The bandwidth is proportional to A/l^2 , which means that there is a physical limit on how much data can be sent over a certain distance given a limited area A . In modern microprocessors, we are close to this limit.

But there are other problems as well. Interconnections (i.e., sending around data on a chip) nowadays take up more than 50% of the total power budget of a microprocessor, and this is rising towards 80% [11]. There are also ecological reasons why we should consider optimizing the power budget of these interconnections: in the US in 2006, the amount of power consumed in datacenters was estimated to be 1.5% of all US electricity [12, 13], and this power consumption approximately doubled in 2011.

Using photonics is the only known physical solution to circumvent this bandwidth limitation. The underlying reason is the very high carrier frequency of optical signals, which, for telecommunication wavelengths is on the order of 300 THz. This means that dielectrics can be used to guide the waves, which have very low losses. For example: in optical fibers, a fascinating bandwidth of 70 Tb/s over a single fiber has recently been reported [14], and commercial systems can send up to 500 Gb/s of information through a single fiber over a distance of 700 kilometers. Also on-chip, the bandwidth supported by a dielectric waveguide is much larger than that of a resistive metal wire.

Photonics offers a solution to many other problems for interconnections, of which we only mention a few here: first, the interconnection energy can, in certain circumstances, be lower than its electrical equivalent. Second, with photonics one can create very precise timing in clocks and signals (reducing the need for synchronization circuitry). This is because the degradation (loss, time jitter) of optical signals in dielectric materials is several orders of magnitude smaller than the degradation of electrical signals in metallic wires. Third, there is no electromagnetic interference. As a consequence of all previously mentioned points, the overall design complexity of a chip can be reduced. A detailed description of all possible benefits of optical interconnects can be found

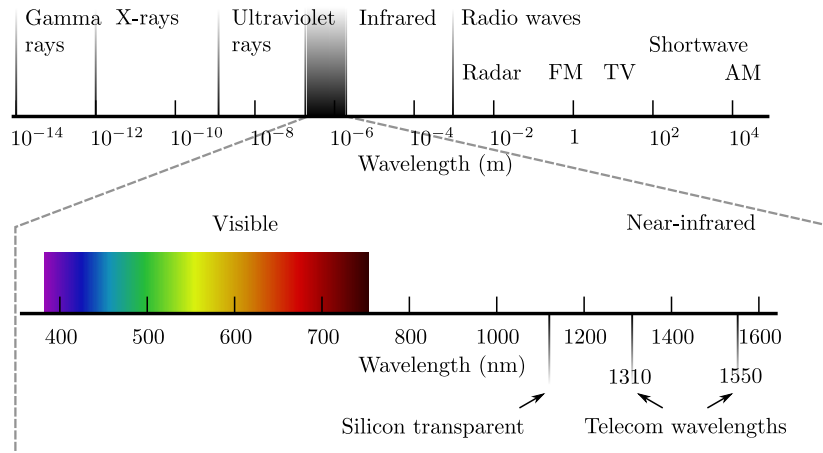


Figure 1.2: The electromagnetic spectrum. The most interesting part for photonics is in the visible to near-infrared window. Silicon becomes transparent around 1110 nm, and two wavelengths often used in telecommunication are wavelengths around 1310 nm and around 1550 nm. Other material systems such as Silicon Nitride operate in the visible region.

in [11].

In the next section, we give a small introduction in the field of photonics, and we then further elaborate on nanophotonics as a platform with great potential for faster and more power-efficient information processing.

1.2 Nanophotonics

In photonics we study the interaction of light with matter. More specifically, the field studies the propagation and the generation of light in different media such as dielectrics, air and metals.

Photonics has many applications such as sensing (gas sensing, biosensing), telecommunication, lighting, photovoltaics, CD/DVD drives and so on. For most photonic applications, the wavelengths of interest are between the visible and the infrared, as shown in Figure 1.2.

A recent trend in photonics is the drive towards miniaturization of components and integrating many of them on a single chip. These so-called (nano)photonic integrated circuits have a better performance, are cheaper, are more robust, and consume less power than bulk photonics, than low-contrast integrated photonics and than electronics.

One excellent material for guiding light is silicon. Silicon has very low losses

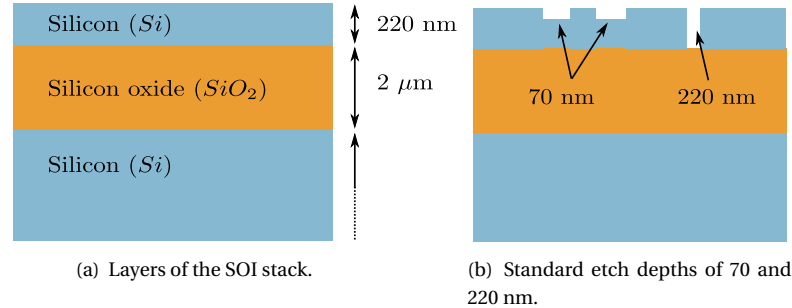


Figure 1.3: The layers of a standard SOI stack. The thickness of the bottom Silicon layer depends on the way the SOI stack was fabricated and whether or not the final wafer is thinned. The top layer is patterned to create nanophotonic structures. Typically, etch depths of 70 and 220 nm are used.

in wavelengths that are useful for telecommunication (1310 nm and 1550 nm), and thanks to the high index contrast, we can produce very small devices. To make nanophotonic chips, typically one starts from a Silicon On Insulator (SOI) wafer, see Figure 1.3(a). Using different resists and etching processes, the wafer is then patterned (see Figure 1.3(b)).

1.2.1 Fabrication of nanophotonic chips

There are essentially two ways to define optical features on-chip. Both methods are based on a resist that covers the chip. Part of the resist is then removed, and in a next step, the unprotected parts of the chip can be etched, or other materials can be deposited on top of it. The first method for modifying the resist is by using electron beam lithography. In electron beam lithography (often abbreviated as e-beam lithography), a beam of electrons is incident on the resist in order to remove parts of it. For example: photonic wire waveguides are fabricated in [15, 16] and photonic crystal cavities are fabricated in [17, 18]. Even though e-beam steering can allow accurate dimensional control, it is slow and unsuitable for mass production due to the small writing area. Figure 1.4 shows some examples of nanophotonic components.

The second way to define optical features is by using a resist that is sensitive to light (a photoresist). Photoresists are used a lot in electronic chip fabrication, and moreover silicon is a good material to guide light, so we can reuse standard Complementary Metal Oxide Semiconductor (CMOS) technology to manufacture photonic chips. In this technology, the SOI wafer is patterned using deep UV lithography. Recent advances in the lithography processes made it possi-

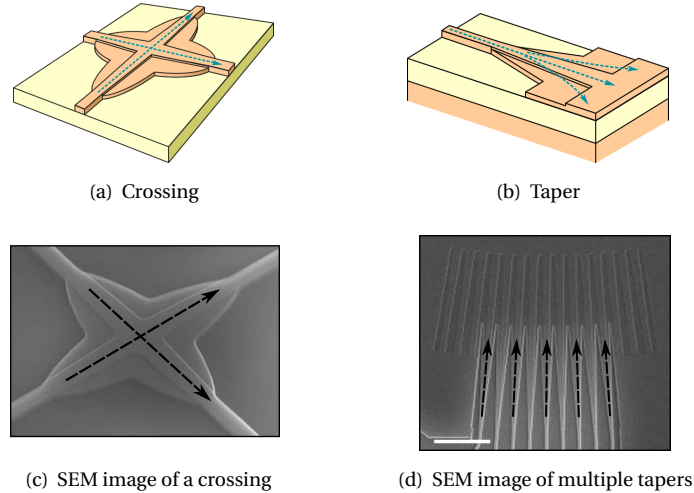


Figure 1.4: Some examples of nanophotonic subcomponents created by optical lithography. These components are building blocks for integrated optical circuits. Because a nanophotonic circuit is planar, crossings (left) are sometimes needed. Tapers (right) are used to spread light from a narrow waveguide to a broad one. On the bottom, Scanning Electron Microscope (SEM) pictures of the fabricated devices are shown.

ble to accurately pattern optical structures with a dimensional control of 1-5 nm [19], which enables mass-fabrication of nanophotonic devices and circuits. A detailed step-by-step description of the process we use at imec can be found in [19].

One big challenge when it comes to guiding light on a chip, is to control the light in a very accurate manner. The features that exhibit guiding properties are only sub-micron scale (e.g., 450 nm thick and 220 nm high for a rectangular waveguide), and the phase of the light is very sensitive to slight variations in these dimensions. Surface roughness causes scattering and back reflections, which lead to more losses and performance degradation. We will discuss the impact of this precision on the performance of our devices in chapter 3 and 6.

1.2.2 Nonlinearities

Nonlinear processes cause a change in the refractive index n of the material. Depending on the used materials and the type of nonlinearity, the strengths of the effects can vary over different orders of magnitude. Furthermore, the timescale at which the different nonlinear effects occur varies from the microsecond (μs)

scale to the femtosecond ($f s$) scale.

The fastest nonlinearity that we will encounter in this dissertation is the Kerr effect. The Kerr effect causes the refractive index to change in response to an applied electric field. This can be either an externally applied field or the optical field itself. In the latter case, $n = n_0 + n_2 I$, where I is the intensity of the light and n_2 is the Kerr constant. As n_2 is usually very small, this effect is only relevant for high intensities. For silicon, n_2 is on the order of $10^{-13} cm^2/W$ for telecom wavelengths [20] (which is still a factor 200 higher than in silicon oxide). In resonant structures, the Kerr effect can cause a bistability of the output, and very interesting nonlinear behavior arises when coupling several of these cavities.

One of the other important nonlinearities that has to be taken into account is the temperature effect. Due to temperature changes caused by high optical powers or resistive heating, the refractive index can vary according to $n = n_0 + \frac{dn}{dT} \Delta T$. For silicon, $\frac{dn}{dT} \simeq 1.86 \cdot 10^{-4} K^{-1}$ [21, 22]. Sometimes heaters are positioned on top of the optical structures in order to control the refractive index, for example in nanophotonic beam steering [23, 24].

In nanophotonic resonators such as photonic crystal cavities and ring resonators, the intensity inside the resonating structure can become very high, meaning that nonlinear effects will play a more important role. In addition these resonant devices are very sensitive to phase changes caused by refractive index changes. We can estimate the wavelength shift $\Delta \lambda$ using the following equation:

$$\frac{\Delta \lambda}{\lambda} = \frac{\Delta n}{n}, \quad (1.1)$$

where Δn is the change in refractive index. In the case of a thermal effect, for $\lambda \simeq 1550$ nm and $n \simeq 3$ and a temperature increase of ten degrees, this results in a significant shift of about 1 nm.

1.2.3 Building blocks for optical neural networks

As we explained previously, nanophotonics could be used as a platform to create an artificial neural network. Artificial neural networks, which we will discuss in more detail in chapter 2, consist of a large number of nonlinear elements that are connected to each other and together perform computation. There are several potential building blocks to consider when designing such a nanophotonic neural network. Semiconductor Optical Amplifiers (SOAs) have been extensively investigated in the doctoral thesis of K.T. Vandoorne [8], and ring resonators are being investigated by T. Van Vaerenbergh, see for example [25, 26]. Another interesting class of components are photonic crystal cavities, which are the emphasis of this doctoral thesis. The main differences between photonic crystal cavities and the other devices are:

- Photonic crystal cavities are passive devices (as opposed to SOAs, which consume approximately 1 mW per SOA). If the insertion loss (IL) of a cavity is sufficiently low, we do not need much regeneration of the signal in the network, leading to low-power reservoirs.
- Cavities store energy in a cavity mode. This leads to a considerable build-up of energy, which causes nonlinear effects such as temperature effects, the plasma dispersion effect due to free carriers, and Kerr-nonlinearities to become present. This is an advantage when a reservoir task needs non-linearity. Furthermore, the cavity has a time constant, which is a memory mechanism that is similar to that in leaky hyperbolic tangent reservoirs. By playing with the dimensions of the device, we can modify this time constant.
- The resonance mechanism for a ring resonator and a photonic crystal cavity are very similar. However, a photonic crystal cavity is inherently bidirectional. This can be advantageous because this adds additional feedback paths into the system.
- Photonic crystal cavities can have two possible architectures. Either the cavity is put within the guiding structure (inline cavity), which means that the transmission only equals unity when the device is at resonance. In the other case, the cavity is next to the waveguide (side cavity), and the transmission equals zero at resonance. This makes them behave fundamentally different, and gives additional degrees of freedom when designing a reservoir.
- The cavity can be made to be very compact, which means we can put a large number of cavities on one chip. This is especially true for 1D wire cavities (which are discussed in section 3.3), which can be fabricated with a very small footprint of about 10-20 μm using SOI technology. Ring resonators on the other hand, with bend radii of 5 μm and larger, have a footprint of at least 100 μm^2 , and SOAs, with a length of at least 500 μm , are relatively large compared to the other two devices.

1.3 Goal

The goal of this dissertation is twofold.

First, we assess whether we can design a nanophotonic reservoir computer based on passive photonic crystal cavities, which we can train to perform computation. We want to train this novel architecture for the speech task, where we use a conventional off-line learning rule. Additionally we investigate whether

we can successfully use an on-line learning rule and train the network to generate periodic signals. Moreover, we identify and study the design challenges for fabricating a passive nanophotonic reservoir.

Second, we provide the researchers in nanophotonic reservoir computing with a framework with which they can model current and future optical circuits. It has to be fast and flexible, and it has to allow the simulation of different topologies easily. Furthermore, it needs to have an efficient way to add new building blocks. This has resulted in the circuit simulator called Caphe⁵, which turned out to be useful not only for reservoir computing applications, but for the nanophotonic research field in general.

1.4 Thesis outline

This thesis is structured as follows: in chapter 2, we introduce the field of reservoir computing, and discuss the learning methods which we will use throughout this dissertation.

Chapter 3 then deals with the detailed modeling of photonic crystal cavities, the nanophotonic building block which we use for constructing the reservoir.

Then, in chapter 4, we discuss the mathematical framework which we have developed during this thesis, where we use high-level approximated mathematical models to simulate large networks of nonlinear nanophotonic components.

Afterwards, we will perform simulation experiments on two different tasks. The first task is the speech recognition task, discussed in chapter 5. In this chapter, we train a nanophotonic reservoir of photonic crystal cavities so it can classify isolated spoken digits.

The second task is a signal generation task, where the same reservoir is simulated with a feedback loop, and where the reservoir is trained to generate periodic patterns.

Chapter 7 finally deals with the main conclusions and future perspectives.

1.5 Publications

Publications in international journals

1. M. Fiers, E. Lambert, S. Pathak, P. Dumon, B. Maes, P. Bienstman, W. Bogaerts, *Improving the design cycle for nanophotonic components*, submitted for publication in Journal of Computational Science, (accepted).

⁵The name originally refers to CAvity PHENomenological modeling framework, much like CAvity Modeling Framework CAMFR, developed by my promoter Peter Bienstman.

2. T. Van Vaerenbergh, M. Fiers, J. Dambre, P. Bienstman, *Simplified description of self-pulsation and excitability by thermal and free-carrier effects in semiconductor microcavities*, Physical Review A, 86(6), p.063808 (2012)
3. M. Fiers, T. Van Vaerenbergh, F. Wyffels, D. Verstraeten, B. Schrauwen, J. Dambre, P. Bienstman, *Generating Periodic Patterns Using Optical Resonators*, submitted for publication in IEEE Transactions on Neural Networks and Learning Systems.
4. T. Van Vaerenbergh, M. Fiers, P. Mechet, T. Spuesens, R. Kumar, G. Morthier, B. Schrauwen, J. Dambre, P. Bienstman, *Cascadable Excitability in microrings*, Optics Express, 20(18), p.20292-20308 (2012)
5. M. Fiers, T. Van Vaerenbergh, K. Caluwaerts, D. Vande Ginste, B. Schrauwen, J. Dambre, P. Bienstman, *Time-domain and frequency-domain modeling of nonlinear optical components on circuit-level using a node-based approach*, Journal of the Optical Society of America B, 29(5), p.896-900 (2012)
6. E. Lambert, M. Fiers, S. Nizamov, M. Tassaert, S. G. Johnson, P. Bienstman, W. Bogaerts, *Python bindings for the open source electromagnetic simulator MEEP*, Computing in Science and Engineering, 13(3), p.53-65 (2011)
7. B. Maes, M. Fiers, P. Bienstman, *Self-pulsing and chaos in short chains of coupled nonlinear microcavities*, Physical Review A, 80, p.033805 (2009)

Publications in international conferences

1. W. Bogaerts, Y. Li, S. Pathak, A. Ruocco, M. Fiers, A. Ribeiro, E. Lambert, P. Dumon, *Integrated design for integrated photonics: from the physical to the circuit level and back (invited)*, SPIE Optics and Photonics, Czech Republic, (to be published)
2. W. Bogaerts, P. Dumon, M. Fiers, A. Ribeiro, M. Vanslembrouck, *Silicon photonics integrated design*, Fiber Optics and Photonics 2012 (invited), India, p.M.2.B.1 (2012)
3. T. Van Vaerenbergh, M. Fiers, P. Mechet, T. Spuesens, R. Kumar, G. Morthier, K.T Vandoorne, B. Schneider, B. Schrauwen, J. Dambre, P. Bienstman, *Characterization of cascadable excitability in a silicon-on-insulator microring*, Proceedings of the 2012 Annual Symposium of the IEEE Photonics Society Belenux Chapter, Belgium, p.119-122 (2012)
4. W. Xie, M. Fiers, S. Selvaraja, J. Van Campenhout, P. Absil, D. Van Thourhout, *High-Q photonic crystal nanocavities on 300 nm SOI substrate fabricated*

- by 193 nm immersion lithography*, Proceedings of the 2012 Annual Symposium of the IEEE Photonics Society Benelux Chapter, Belgium, p.183-186 (2012)
5. P. Bienstman, K.T Vandoorne, T. Van Vaerenbergh, M. Fiers, B. Schneider, D. Verstraete, B. Schrauwen, J. Dambre, *Reservoir computing on nanophotonic chips*, 2012 International symposium on Nonlinear Theory and its Applications (NOLTA 2012), Spain, p.506-508 (2012)
 6. T. Van Vaerenbergh, M. Fiers, P. Mechet, T. Spuesens, R. Kumar, G. Morthier, K.T Vandoorne, B. Schneider, B. Schrauwen, J. Dambre, P. Bienstman, *Self-pulsation and excitability mechanism in silicon-on-insulator microrings*, 2012 Asia Communications and Photonics Conference (ACP) (2012)
 7. W. Bogaerts, P. Dumon, E. Lambert, M. Fiers, S. Pathak, A. Ribeiro, *IPKISS: A Parametric Design and Simulation Framework for Silicon Photonics*, 9th International Conference on Group IV Photonics, United States, p.30-32 (2012)
 8. M. Fiers, K.T Vandoorne, T. Van Vaerenbergh, J. Dambre, B. Schrauwen, P. Bienstman, *Optical Information Processing: Advances in Nanophotonic Reservoir Computing*, International Conference on Transparent Optical Networks (invited), United Kingdom, (2012)
 9. M. Fiers, T. Van Vaerenbergh, K. Caluwaerts, J. Dambre, P. Bienstman, *CAPHE: Time-domain and Frequency-domain Modeling of Nonlinear Optical Components*, Advanced Photonics Congress, 2012 OSA, United States, p.paper IM2B.3 (2012)
 10. T. Van Vaerenbergh, M. Fiers, K.T Vandoorne, B. Schneider, J. Dambre, P. Bienstman, *Towards a photonic spiking neuron: excitability in a silicon-on-insulator microring*, accepted for publication in NOLTA, (to be published).
 11. K.T Vandoorne, T. Van Vaerenbergh, M. Fiers, P. Bienstman, D. Verstraeten, B. Schrauwen, J. Dambre, *Photonic reservoir computing and information processing with coupled semiconductor optical amplifiers*, Fifth 'RIO DE LA PLATA' Workshop on Laser Dynamics and Nonlinear Photonics (invited), Uruguay, p.1-3 (2011)
 12. M. Fiers, T. Van Vaerenbergh, P. Dumon, K. Caluwaerts, B. Schrauwen, J. Dambre, P. Bienstman, *CAPHE: a circuit-level time-domain and frequency-domain modeling tool for nonlinear optical components.*, 16th Annual Symposium of the IEEE Photonics Benelux Chapter, (2011)

13. K.T Vandoorne, M. Fiers, T. Van Vaerenbergh, D. Verstraeten, B. Schrauwen, J. Dambre, P. Bienstman, *Advances in photonic reservoir computing on an integrated platform*, International Conference on Transparent Optical Networks (ICTON) (invited), Sweden, p.Mo.B5.5 (2011)
14. K.T Vandoorne, M. Fiers, D. Verstraeten, B. Schrauwen, J. Dambre, P. Bienstman, *Optical signal processing with a network of semiconductor optical amplifiers in the context of photonic reservoir computing*, SPIE Photonics West - OPTO, 7942, United States, p.79420P (2011)
15. E. Lambert, M. Fiers, W. Bogaerts, D. Vermeulen, P. Bienstman, *A Python Software Framework for the Design of Photonic Integrated Circuits*, EuroScipy 2010, France, (2010)
16. K.T Vandoorne, M. Fiers, D. Verstraeten, B. Schrauwen, J. Dambre, P. Bienstman, *Photonic Reservoir Computing: a New Approach to Optical Information Processing*, Workshop on "Cognitive and neural models for automated processing of speech and text" (CONAS), Belgium, (2010)
17. K.T Vandoorne, M. Fiers, D. Verstraeten, B. Schrauwen, J. Dambre, P. Bienstman, *Photonic Reservoir Computing: a New Approach to Optical Information Processing*, International Conference on Transparent Optical Networks 2010 (ICTON) (invited), Germany, p.Th.A4.3 (2010)
18. P. Bienstman, K.T Vandoorne, M. Fiers, D. Verstraeten, J. Dambre, B. Schrauwen, *Photonic reservoir computing as a new paradigm for optical information processing*, Photonics North 2010 (invited), Canada, p.196 (paper PDS-2-1-4) (2010)
19. M. Fiers, B. Maes, P. Bienstman, *Dynamics of coupled cavities for optical reservoir computing*, IEEE Photonics Benelux symposium, Belgium, p.129-132 (2009)
20. B. Maes, M. Fiers, P. Bienstman, *Self-pulsing and chaos in networks of nonlinear resonators*, 18th International Laser Physics Workshop (LPHYS) (invited), Spain, p.345 (2009)
21. B. Maes, M. Fiers, K. Huybrechts, G. Morthier, P. Bienstman, *Dynamics and Instabilities in Series of Coupled Nonlinear Resonators*, ICTON 2009 (invited)

Publications in national conferences

1. T. Van Vaerenbergh, M. Fiers, K.T Vandoorne, J. Dambre, P. Bienstman, *Photonic Reservoir Computing using thermal nonlinearities in microrings*, 12th FEA PhD Symposium (2011)

References

- [1] N. Cristianini and J. Shawe-Taylor. *An introduction to support vector machines and other kernel-based learning methods*. Cambridge university press, 2000.
- [2] David A. Drachman. *Do we have brain to spare?* *Neurology*, 2004-2005.
- [3] W. Gerstner and W. Kistler. *Spiking Neuron Models: Single Neurons, Populations, Plasticity*. Cambridge University Press, 2002.
- [4] <http://www.top500.org/>.
- [5] L. Appeltant, M.C. Soriano, G. Van der Sande, J. Danckaert, S. Massar, J. Dambre, B. Schrauwen, C.R. Mirasso, and I. Fischer. *Information processing using a single dynamical node as complex system*. *Nature Communications*, 2011.
- [6] Kristof Vandoorne, Wouter Dierckx, Benjamin Schrauwen, David Verstraeten, Roel Baets, Peter Bienstman, and Jan Van Campenhout. *Toward optical signal processing using Photonic Reservoir Computing*. *Optics Express*, 16(15):11182–11192, JUL 21 2008.
- [7] Kristof Vandoorne, Joni Dambre, David Verstraeten, Benjamin Schrauwen, and Peter Bienstman. *Parallel reservoir computing using optical amplifiers*. *IEEE Transactions On Neural Networks*, 22(9):1469–1481, 2011.
- [8] Kristof Vandoorne. *Photonic Reservoir Computing with a Network of Coupled Semiconductor Optical Amplifiers*. PhD thesis, Ghent University, 2011-2012.
- [9] David Sussillo and L. F. Abbott. *Generating Coherent Patterns of Activity from Chaotic Neural Networks*. *Neuron*, 63(4):544–557, AUG 27 2009.
- [10] D.A.B. Miller and H.M. Ozaktas. *Limit to the bit-rate capacity of electrical interconnects from the aspect ratio of the system architecture*. *Journal of Parallel and Distributed Computing*, 41(1):42–52, 1997.
- [11] D. Miller. *Device requirements for optical interconnects to silicon chips*. *Proceedings of the IEEE*, 97(7):1166–1185, 2009.
- [12] D.A.B. Miller. *Are optical transistors the logical next step?* *Nature Photonics*, 4(1):3–5, 2010.
- [13] http://www.energystar.gov/ia/partners/prod_development/downloads/EPA_Datacenter_Report_Congress_Final1.pdf.

- [14] Akihide Sano, Hiroji Masuda, Takayuki Kobayashi, Masamichi Fujiwara, Kengo Horikoshi, Eiji Yoshida, Yutaka Miyamoto, Munehiro Matsui, Masato Mizoguchi, Hiroshi Yamazaki, Yohei Sakamaki, and Hiroyuki Ishii. *69.1-Tb/s (432 x 171-Gb/s) C- and Extended L-Band Transmission over 240 Km Using PDM-16-QAM Modulation and Digital Coherent Detection*. In Optical Fiber Communication Conference, page PDPB7. Optical Society of America, 2010.
- [15] M. Gnan, S. Thorns, DS Macintyre, RM De La Rue, and M. Sorel. *Fabrication of low-loss photonic wires in silicon-on-insulator using hydrogen silsesquioxane electron-beam resist*. Electronics Letters, 44(2):115–116, 2008.
- [16] Y.A. Vlasov, S.J. McNab, et al. *Losses in single-mode silicon-on-insulator strip waveguides and bends*. Opt. Express, 12(8):1622–1631, 2004.
- [17] Eiichi Kuramochi, Masaya Notomi, Satoshi Mitsugi, Akihiko Shinya, Takasumi Tanabe, and Toshifumi Watanabe. *Ultra-high-Q photonic crystal nanocavities realized by the local width modulation of a line defect*. Applied Physics Letters, 88(4):041112, 2006.
- [18] M Devittorio. *Two-dimensional photonic crystal waveguide obtained by e-beam direct writing of SU8-2000 photoresist*. Microelectronic Engineering, 73-74:388–391, 2004.
- [19] Shankar Kumar Selvaraja, Patrick Jaenen, Wim Bogaerts, Dries Van Thourhout, Pieter Dumon, and Roel Baets. *Fabrication of Photonic Wire and Crystal Circuits in Silicon-on-Insulator Using 193-nm Optical Lithography*. J. Lightwave Technol., 27(18):4076–4083, Sep 2009.
- [20] A.D. Bristow, N. Rotenberg, and H.M. Van Driel. *Two-photon absorption and Kerr coefficients of silicon for 850–2200*. Applied physics letters, 90(19):191104–191104, 2007.
- [21] Bradley J. Frey, Douglas B. Leviton, and Timothy J. Madison. *Temperature-dependent refractive index of silicon and germanium*. arxiv.org/pdf/physics/0606168, 2006.
- [22] F. De Leonardis, A. V. Tsarev, and V. M. N. Passaro. *Optical properties of new wide heterogeneous waveguides with thermo optical shifters*. Optics Express, 16(26):21333–21338, 2008.
- [23] K. Van Acoleyen, E. Ryckeboer, W. Bogaerts, and R. Baets. *Efficient Light Collection and Direction-of-Arrival Estimation Using a Photonic Integrated Circuit*. Photonics Technology Letters, IEEE, 24(11):933–935, 2012.

-
- [24] K. Van Acoleyen, W. Bogaerts, J. Jágerská, N. Le Thomas, R. Houdré, and R. Baets. *Off-chip beam steering with a one-dimensional optical phased array on silicon-on-insulator*. *Optics letters*, 34(9):1477–1479, 2009.
- [25] T. Van Vaerenbergh, M. Fiers, P. Mechet, T. Spuesens, R. Kumar, G. Morthier, B. Schrauwen, J. Dambre, and P. Bienstman. *Cascadable Excitability in microrings*. *Optics Express*, 20(18):20292–20308, 2012.
- [26] T. Van Vaerenbergh, M. Fiers, J. Dambre, and P. Bienstman. *Simplified description of self-pulsation and excitability by thermal and free-carrier effects in semiconductor microcavities*. *Physical Review A*, 86(6):063808, 2012.

2

Reservoir Computing

*“I think I can say fairly that Deep Blue did not destroy chess. There was perhaps even a mini boom in chess popularity as a result of the Deep Blue matches.”
(Feng-hsiung Hsu, one of the main programmers of Deep Blue, after Deep Blue’s 1997 match win over World Chess Champ Garry Kasparov).*

As we have mentioned in the introductory chapter, there are several ways to process information. On one side, there’s the conventional computer, which is good at processing large amounts of data, following a predefined step-by-step algorithm. However, there is a wide variety of tasks which humans typically find easy to solve, but where the solution cannot be described adequately in an algorithmic way. Typically these are tasks where one wants to understand the nature of the input, and where one wants to be able to generalize, i.e., respond to unseen input, as is the case for speech and image recognition, robot locomotion and so on. It is not surprising that there exist artificial systems inspired by the human brain, that perform much better on these tasks. A particular brand of so-called artificial neural networks are the topic of this chapter.

This chapter is structured as follows: in section 2.1 we first give an introduction to these two forms of computation, i.e., conventional versus brain-inspired computation. In section 2.2 we give an introduction to the field of machine learning. A decade-old technique called Reservoir Computing (RC) makes it

particularly easy to perform training, and is routinely used in this dissertation. Therefore, we give a more rigorous introduction to Reservoir Computing in section 2.3.

This chapter does not aim to be a comprehensive discussion on machine learning methods nor a rigorous proof of mathematical properties of these systems. Instead, we want to give an overview of the main ideas behind machine learning and reservoir computing, from a practical and engineering point of view. The doctoral thesis of D. Verstraeten [1] provides a more complete overview of state-of-the-art techniques and contains in-depth information on the history, mathematics, comparison with other techniques, different performance evaluation methods and so on.

2.1 Information processing

The theoretical framework described by Alan Turing [2] formed the basis for some first discrete-symbol computers, and as such it forms the basis for most modern systems able to perform computation. In his work, Alan Turing presented an abstract device, the *Turing machine*, that describes how algorithms can be executed. It consists of a processing unit (the head) that contains a program which defines its behavior. The head operates on a separate storage device for symbols (the tape). This architecture can be rendered more general by storing the program in the same memory as the data. This architecture, called the *universal Turing machine* can simulate any given Turing machine.

An important property of the universal Turing machine is that the behavior of its processing unit is explicitly programmed. Moreover, it is step-based, and it reads discrete symbols, not continuous values. The popularity of this architecture is due to various technological and historical reasons, and until today is the most common way to perform computation.

This architecture was later adopted by Von Neumann, who describes an electronic digital computer. This computer has a Central Processing Unit (CPU), memory that stores both the programs and data, an external storage, and input-output mechanisms. Almost all personal computers nowadays are based on this architecture¹.

Standard computer programs (that are executed on a Von Neumann-based machine) are very good at processing large amounts of predictable data in a

¹The term von Neumann architecture arose from von Neumann's paper, "First Draft of a Report on the EDVAC", dating from June 1945. It was unfinished when his colleague Herman Goldstine circulated it with only von Neumann's name on it. Credit should go to J. Presper Eckert and John Mauchly for their contributions as well.

Alan Turing cites the First Draft in "Proposals for Development in the Mathematics Division of an Automatic Computing Engine (ACE)," presented to the National Physical Laboratory, 1945, as *the definitive source for understanding the nature and design of a general-purpose digital computer*.

deterministic way. However, it is not the only architecture that is capable of performing computation. The human brain, as a best example, is not an explicitly programmed, discrete, step-by-step system. And yet it is very efficient in a range of tasks that a computer cannot solve easily, such as speech and image recognition, controlling the motion of a robot arm, medical diagnosis and so on. What makes our brain so unique? Most of it is due to the fact that our brain can learn by example, based on a limited dataset, and then generalize to solve the problem with unseen input. It thereby “understands” the nature of the problem, whereas the computer typically cannot.

These are the typical problems that are investigated in the research fields of Artificial Intelligence and Machine Learning. Artificial Intelligence is a very broad research field, and involves general problems such as how to define intelligence (see for example Alan Turing’s test). It investigates reasoning, perception, communication, knowledge and so on. Machine Learning (ML) techniques are used to solve many of the problems that are investigated in AI. One important concept in this field is the Artificial Neural Network (ANN). These systems appear to be computationally very rich, although it is not trivial to train such a system. Artificial neural networks, when trained properly, outperform traditional algorithms in a wide variety of tasks: speech recognition, robot control and localization, epileptic seizure detection, brain computer interfaces, handwriting recognition, financial forecasting, attractor learning, tunable pattern generators and many more.

A lot of research effort goes into efficiently training neural networks. Some of these training methods are explained later in this chapter and give us a better understanding of why these methods work so well. There is also a research effort to try and make the neural network as large as possible. Just as an illustration, we list some of these extraordinary large artificial neural networks.

The SPAUN The Semantic Pointer Architecture Unified Network [3] (SPAUN) consists of 2.5 million virtual neurons. It is capable of remembering a sequence of digits and predicting the next logical digit, almost in the same way as humans do, but it takes one hour to simulate one second of neural behavior².

A Cat’s brain In the paper “The Cat is Out of the Bag: Cortical Simulations with 10^9 Neurons, 10^{13} Synapses” (2009), Rajagopal Ananthanarayanan and his colleagues at IBM describes how conventional supercomputers can perform a

²Raymond Kurzweil predicts that within a few decades, machine-intelligence will exceed human intelligence. This *technological singularity*, according to futurists such as Kurzweil, will be possible thanks to the exponential growth of processing capacity, as predicted by Moore’s law.

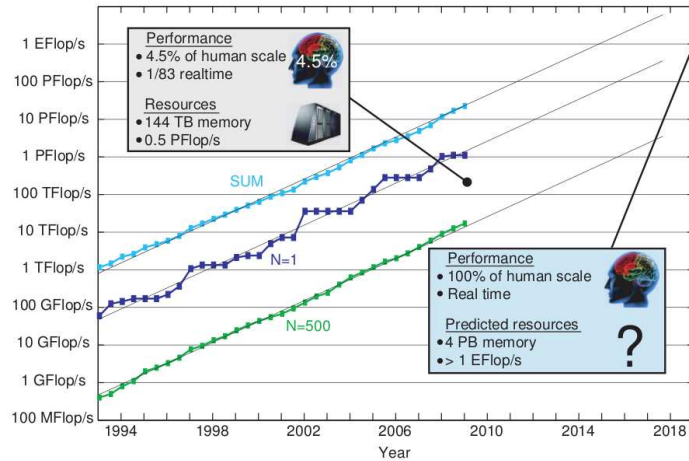


Figure 2.1: Image from "The Cat is Out of the Bag: Cortical Simulations with 10^9 Neurons, 10^{13} Synapses". Growth of the Top 500 supercomputers (N=1 is the strongest computer, and SUM is the sum of all computing power) overlaid with the results from the paper and a projection for realtime human-scale cortical simulation.

full cortical simulation of the brain of a cat³. Figure 2.1 shows how they predict that real-time human-scale simulations are inevitable by the year 2018.

SpiNNaker The SpiNNaker Project, involving several UK-based universities and companies, including ARM, is developing a hardware platform based on the ARM chip, targeting to combine 50,000 chips to create 1 billion of simple computing elements. It is different than the other approaches, because the focus of this project is to make an efficient implementation of a brain without explicitly using the Von Neumann architecture. Instead, these very simple computing elements look very similar to the neurons of an artificial neural network.

The Human Brain Project This large-scale European project⁴ is another project that is aimed towards understanding the human brain. Its first goal is to integrating several ICT-based research platforms, one of which is a Neuromorphic Computing Platform, which makes it possible to translate brain models into a new class of hardware devices and to test their applications.

³This just illustrates the processing capacity of current supercomputers. This certainly does not mean that we can now manufacture a robot cat or simulate a virtual cat on your desktop (for example, this system does not yet get input from virtual sensing organs such as vision, touch and smell, nor does it simulate any muscle or organ).

⁴<http://www.humanbrainproject.eu/>

2.2 Machine Learning

The goal in Machine Learning is to create a system that, for a certain task, tries to discover how the desired output is related to its inputs. This should be done in a robust way, such that the system is able to respond to input it has not seen before. Instead of mapping the input explicitly to the output, the system has learned in some way, e.g. by generalizing from training examples. Fundamental work on statistical learning theory [4] and neural networks [5] has made the field of ML grow explosively. The combination of increasing computational power and theoretical progress in understanding the nature of learning, has made machine learning into a well-established and respected research area.

2.2.1 Different learning methods

Basically, there are three classes of ML methods based on the way they implement learning:

- **Unsupervised learning methods:** These systems learn only from the examples presented to them. They have no clue as to what behavior is required. The main goal of these types of algorithms is to discover regularities or structure in the data without explicitly having to program them. Examples are Self Organizing Maps (SOM) [6] or K-means clustering [7], and applications include data clustering and information retrieval.
- **Reinforcement learning methods:** In these methods, the system receives information about how well it is doing, but it does not receive the exact information of what the output should be. When presented with an example input, the response of the system is evaluated and scored, but the actual correct behavior is never given. These algorithms are popular in the robotics community, as it is often easier to define a reward signal than an explicit training signal. For example, a walking robot can be rewarded for moving faster, and a penalty is given when it falls. The system then learns from the rewards and penalties. This technique is also popular for learning complex games such as Go [8] and Backgammon [9].
- **Supervised learning methods:** This type of learning mechanism relies on a set of known input examples with a given desired output. This set is used for training the system. Although in many real world applications this information is not available, knowing the exact input and output for several samples can greatly accelerate the learning process and lead to superior accuracy.

It is also possible to combine several of the above-mentioned methods. For example, a semi-supervised learning methods combines the first and third

method. In many applications (especially in imaging), collecting label samples (i.e., known input-output pairs) is time-consuming and expensive, so this technique relies on relatively few labeled samples (from which class separation may be inferred), in combination with a large number of unlabeled ones (from which the structure of the data can be discovered).

In this dissertation we focus on supervised learning methods, where we have a priori knowledge of the correct output. Typically, this method involves the minimization of a set of weights. To avoid that these weights become too large (which is typically an indication that the system is overfitting), some sort of regularization is usually performed (this will be further explained in section 2.3.3.1). In one way, this is similar to the principle of Ockham's Razor: simple rules (i.e., rules where the weights are not very large) typically generalize better (because they do not overfit).

As we mentioned before, Artificial Neural Networks (ANN) can be used to solve machine learning problems. These ANNs are attractive because they are biologically plausible (which means they resemble the way our brain is built), and because they can be simulated very fast on current computers. In the next section we explain in more detail what these ANNs are.

2.2.2 Artificial Neural Networks

An Artificial Neural Network (ANN) is essentially a model of the brain structure. It consists of neurons and connections, just like the brain. Most of the time the term Neural Network is used, and from the context it is usually clear whether it involves an artificial or biological network. A simple ANN with a feedforward topology is shown in Figure 2.2.

2.2.2.1 Neuron types

Depending on the application, different models are used in literature. Here we will briefly discuss three neuron categories.

Threshold gates or perceptrons A first category of neurons, based on McCulloch-Pitts neurons, only produces digital output. They are also called threshold gates or perceptrons. Even though these models are very simple, they are universal for computations⁵ with digital input and output, meaning that every Boolean function can be computed by some feedforward neural network with a single hidden layer.

⁵Computationally universal or Turing complete means that they can model any Turing machine.

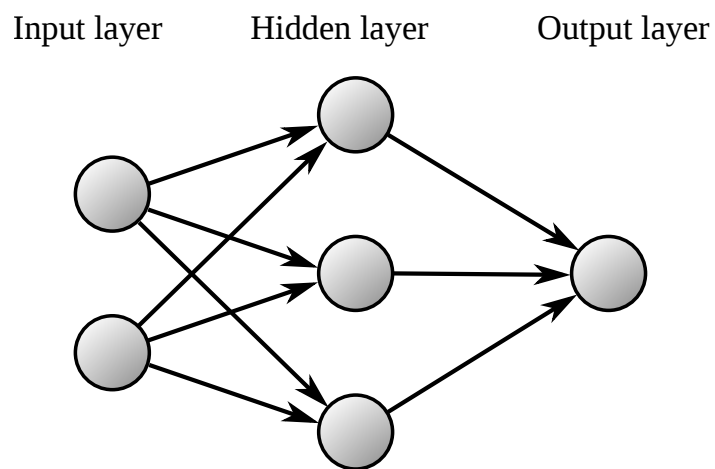


Figure 2.2: A simple artificial neural network. This kind of network is called a feedforward neural network (with one *hidden* layer), since all signals propagate in one direction and there are no feedback connections. A neuron can either perform a weighted sum on its inputs and apply a nonlinear transformation (such as a thresholding function or a sigmoid function), or it can be based on differential equations, as in the case of spiking neurons. In the case of a spiking neural network, the connections (in this case also called synapses) can also embed an exponential filter.

Neurons based on an activation function (analog neurons) A second generation is based on computational units that apply a nonlinear "activation function" with a continuous set of possible output values, based on a weighted sum of the inputs. The most common activation function is a sigmoid-type function such as the logistic or fermi function:

$$\text{fermi}(x) = \frac{1}{1 + \exp(x)}, \quad (2.1)$$

and the tanh function, given by:

$$\tanh(x) = \frac{\exp(x) - \exp(-x)}{\exp(x) + \exp(-x)} = 2\text{fermi}(2x) - 1 \quad (2.2)$$

It is proven that using these neurons, certain boolean functions can be computed with fewer gates than using a NN with threshold gates [10]. A characteristic feature of these neurons is that they are differentiable, which means that they support learning algorithms that are based on gradient descent, such as backpropagation [11, 12].

The input-output relationship of the first two neuron types is given by

$$y_j = f\left(\sum_{i \in S_j} \mathbf{W}_{ij} y_i\right) \quad (2.3)$$

Where y_i is the output or *activation* level of the i th neuron, S_j is a set of indices of neurons with connections leading from neuron i to neuron j , and w_{ij} is the weights between those connections.

Spiking neurons A third type of neurons is commonly referred to as spiking neurons. These are biologically more realistic but they are more complex to model. The resulting networks are called Spiking Neural Networks [13, 14] and it has been shown theoretically that they can perform more complex operations than analog neurons [15]. These neurons communicate through isolated spikes instead of continuous values, and the information is entirely encoded in the precise timing and/or *firing rate* of the neurons. The use of spiking neural networks for engineering applications has been the topic of the PhD of B. Schrauwen [14]. The biggest challenge for these networks is to find a proper way to represent data. Usually, information in the real world is of analog nature, and this has to be translated to isolated events (spikes). There are different ways to encode analog information in spike trains and each method has its advantages and disadvantages. In this thesis we do not investigate spiking neurons, but the interested reader can find a good starting point for further reading in [14]. This does not mean that this architecture is uninteresting for photonics. In fact,

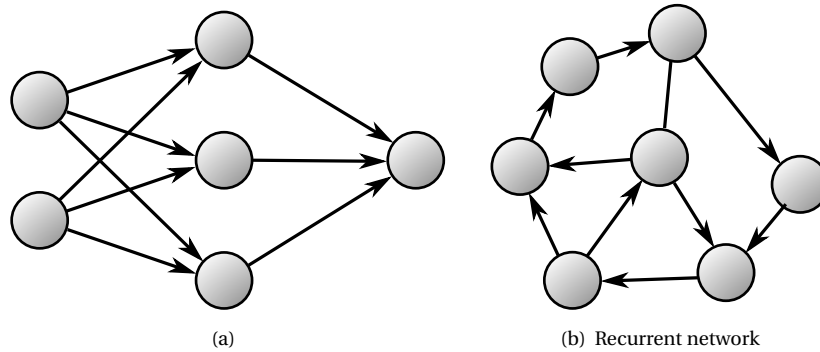


Figure 2.3: Different network topologies for neural networks. (a): a feedforward neural network or multilayer perceptron. (b): a recurrent neural network.

in our research group, the research of T. Van Vaerenbergh is dedicated to simulating and using optical spikes using microring resonators as nanophotonic component for information processing.

In this dissertation we will construct a nanophotonic neural network based on Photonic Crystal Cavities. These cavities can be classified as analog neurons, and we will investigate their activation function in detail in Chapter 3.

2.2.2.2 Network topologies

Neurons are connected to each other through connections. The topology (i.e., the way they are connected) is fully determined by the weight matrix \mathbf{W} , where \mathbf{W}_{ij} determines both the connectivity (a nonzero value means neuron j is connected to neuron i) and the strength -or weight- of the connections. Usually, neural networks are divided into two important sub-classes, depending on their connectivity:

Feedforward Neural Networks (FFNN) or Multi-Layer Perceptrons (MLP) (see Figure 2.3(a)). In a feedforward network there are no recurrent connections. The nodes of the network are divided into different layers, with information flowing from the input layer, through different intermediate layers to the output layer. These intermediate layers are also called *hidden* layers because their activation is usually not measured. One important consequence of the unidirectional propagation is that these networks cannot store temporal information. This means that in most cases these systems are inadequate for temporal problems such as signal generation or speech recognition. Nevertheless there are solutions to this problem, for example by implementing a tapped

delay line into which the samples of the inputs are fed chronologically, where all taps are used as inputs to the network. This is called a Time-Delay Neural Network (TDNN) [16]. The drawback of this approach is that many parameters are needed (more with increasing delays) and furthermore it is a rather artificial way to incorporate time which has no biological analogon.

The best-known training rule is the error-backpropagation rule [5], in which the weights of the connections are repeatedly adjusted so as to minimize a measure of the difference between the actual output vector and the desired output vector.

In general, feedforward neural networks are robust to noise, they can learn by example and generalize the problem, and they have the ability to model highly nonlinear systems.

Recurrent Neural Networks (RNN) (see Figure 2.3(b)). The solution to the shortcoming involving temporal data is to introduce feedback loops in the system, where every connection has a delay. This has an important consequence: the new state of the system not only depends on the inputs, but also on the previous state of the system, and hence on all previous inputs. In this way information stays present in the network for a certain time and is mixed with new information as the simulation progresses. This means that the RNN is ideally suited for solving temporal problems. Applications include the controlling and modelling of complex dynamical systems [17], speech recognition [18, 19], handwriting recognition [20] and so on.

Although it looks like a RNN solves the shortcomings of a FFNN, there are not many learning rules and those that exist are rather involved and converge slowly [21]. Most learning rules use gradients to update the network parameters [5, 22–24], but these become very small after only a few timesteps. There are extensions to these learning rules using second order derivatives of the error surface, but these are very involved [25, 26]. Another problem is that slight changes in initial parameters can have sudden changes in the qualitative behavior of the system, making them dvery difficult to train.

2.2.2.3 Hardware implementations of artificial neural networks

Artificial Neural Networks are simulated on a computer. In practice, this means constructing a connection matrix, setting up a mathematical model of the neurons, and then applying matrix multiplications and nonlinear function transformations to simulate the reservoir step by step. There exists dedicated simulation software for simulating spiking neurons. A graphical processing unit (GPU) can be used in some cases to speed up the simulations.

The translation stage from a neural network model to a conventional computer program that is executed by a microprocessor is far from optimal. While a

computer program essentially runs sequentially, a neural network is inherently a parallel system, where the calculations in all neurons happen at the same time. This is what powers research towards hardware implementations of neural networks. In the electronic domain, research has demonstrated implementations such as an FPGA [27], VLSI [28] and even systems based on a single dynamical node [29]. Also the SpiNNaker project which we encountered in the introduction of this chapter is a good example of a hardware platform in the electronic domain. In the field of nanophotonics, K. Vandoorne has shown using simulations that an integrated circuit of Semiconductor Optical Amplifiers (SOAs) can be used to solve a speech recognition task [30], and in this dissertation we will continue research on nanophotonic neural networks.

A more detailed overview of the existing research towards hardware implementations can be found in [31].

2.3 Reservoir Computing

In this section we describe a relatively new (about a decade old) field of research in machine learning. The Reservoir Computing (RC) method tries to overcome the complex learning issues with recurrent neural networks. The first early descriptions of what is now called Reservoir Computing date from the nineties and can be found in [32, 33]. Both papers describe systems in which a neural network is fixed and left untrained, and only a separate output layer is trained. After these publications, it took some time before the same idea emerged again. In two seminal publications, H. Jaeger [34] and W. Maass [35] independently described a novel training method, which marked the birth of the research field of Reservoir Computing. Later, in 2004, another publication by J. Steil [36] presented similar ideas but from a completely different perspective. In this section we will give a short introduction to the approach of the three authors, and we conclude this last section with a rigorous mathematical description of a RC system. We show how the RNN is constructed and how the output layer is trained, both on- and off-line.

Echo State Network (ESN): In [34], H. Jaeger describes the Echo State Network (ESN). This is a recurrent neural network with random topology and random weights. The weight matrix is globally scaled until the desired dynamical regime is reached. The input is fed into this dynamical network, and the states of the network are then used to train a simple linear regression or classification function. The ESN is very popular from an application point of view, because of its ease of construction and its excellent performance on a variety of difficult benchmark tasks. An ESN has fading memory. This means that, when the system is perturbed, after a while the network returns to a rest state and forgets the

input it has seen. Formally, the network should asymptotically forget its initial state when it is driven by an external signal. An equivalent term used in this context is that the network has the echo state property. The ESN transforms the input to a high-dimensional state space of the neural network. The dynamics should be rich enough, so the linear readout can easily separate different features of the reservoir. However, when the network is too excitable, it can move to a regime where the echo state property no longer holds. The optimal dynamical regime strongly depends on the actual application, and finding the optimal parameters for this regime can be a laborious job of tweaking the global scaling parameter, the input scaling, the type of neuron and so on. Nevertheless, finding an optimal reservoir is purely an optimization task and, in most cases does not involve fundamental roadblocks.

Liquid State Machine (LSM): The Liquid State Machine (LSM), presented by W. Maass [35], is an architecture that shows many similarities with the ESN. Where the ESN is more practical from an engineering point of view and does not aim to be biologically correct, the LSM originates from a research lab active in robotics and neuroscience, where the focus lies in the modeling of neurons in a way that is biologically realistic (spiking neurons). The LSM contains two parts: a dynamical network of spiking neurons, and output neurons that transform the high-dimensional transient states into stable readout values. Two properties are defined as necessary and sufficient conditions in order to perform computation with a LSM machine: a separation property (SP) and an approximation property (AP). The separation property states that different input streams should lead to a separation between the trajectories of internal states⁶. The approximation property states that the output should be able to distinguish and transform different internal states into given target outputs. Whereas the SP depends mostly on the complexity of the liquid, the AP depends mostly on the ability of the readout mechanism to adapt itself to the required task. The simplicity of the readout makes it easy to evaluate the information processing capabilities of the model. The LSM has been used successfully in speech recognition [37] and robotics [38], where a spiking neural network is used to perform the computation. As we discussed in 2.2.2.1, spiking neurons have proven to be more powerful for computation, but it is not trivial to encode/decode temporal, analog data into a train of spikes. Also, it is much more difficult to tune the neurons into the right dynamical regime because of their discontinuous behavior. As a consequence, most of the reservoir computing implementations are of the ESN type.

⁶In the case where you consider the LSM to be a real physical liquid, the separation property could reflect that different perturbations give rise to different wave patterns in the liquid.

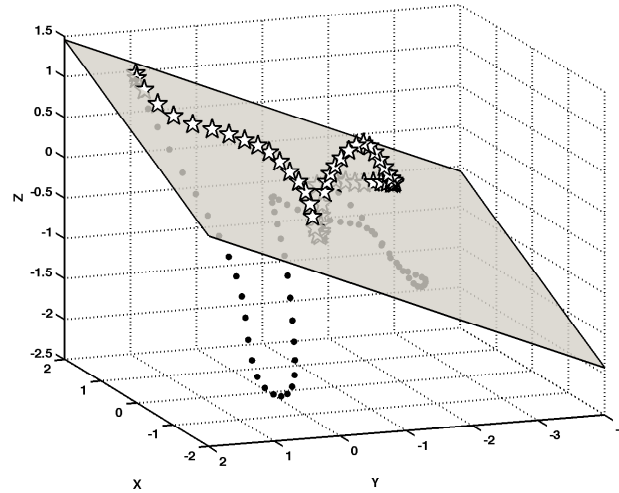


Figure 2.4: By mapping the original feature space to a higher dimensional feature space, it becomes easier to separate them with a linear plane (hyperplane). In this example, time traces of two (different) spoken digits are shown, which represent the state of the reservoir. Here, it is clear that the constructed linear plane can separate the two digits relatively well (picture courtesy of D. Verstraeten).

BackPropagation DeCorrelation (BPDC): In BackPropagation DeCorrelation (BPDC) [36], a network is trained on-line, i.e., the weights are changed during the simulation while new samples are fed to the network. The BPDC rule adapts only the weights of the output neurons where the rest of the RNN is kept fixed. It aims at temporal decorrelation of the network activations w.r.t. the inputs and one-step backpropagated errors, such that the output neuron can optimally read out from the dynamical reservoir.

Reservoir Computing A common idea behind all these approaches is that these systems transform the input signal to a high-dimensional state of the so-called *reservoir* (here, it is just a synonym for the RNN), which makes it easier to extract relevant features. The readout neurons are then trained to extract these features, whereby we rely on well-known and tested methods such as linear regression and regularization. As an illustration, Figure 2.4 shows a (reduced) phase space of a reservoir that processes two spoken digits. It also shows how a two dimensional plane can easily separate the two spoken digits.

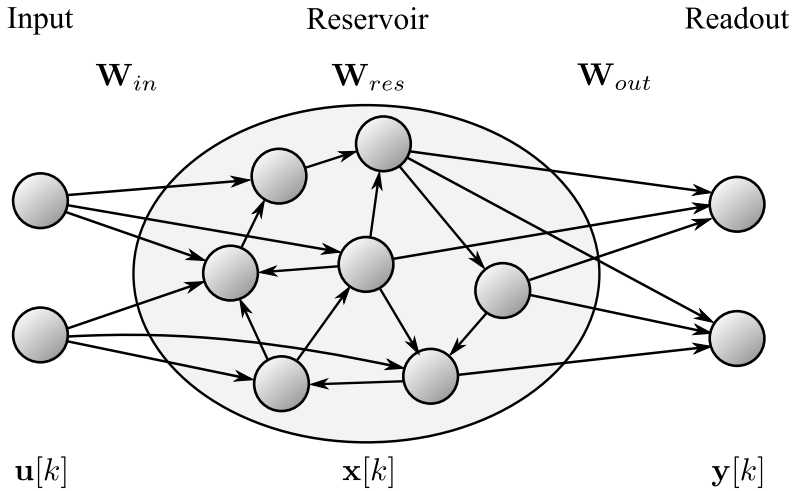


Figure 2.5: A classical discrete-time Reservoir Computing (RC) system. It consists of a Recurrent Neural Network (RNN), called the reservoir, an input layer and a readout layer. The reservoir weights \mathbf{W}_{res} are usually chosen randomly (but globally scaled to reach the desired regime), and are unmodified. Input $\mathbf{u}[k]$ is fed to the reservoir and excites the dynamical system. Features of the dynamical system can be extracted by the linear readout layer. Using a set of known training inputs and desired outputs, the output weights \mathbf{W}_{out} are trained in order to minimize the difference between the actual output and the desired output.

Multiple readouts can be trained to extract different features. For example, in a robot there are different joints that have to be controlled, and different output neurons can be trained to steer different muscles.

By far the most involving part of setting up a RC system is to tune the reservoir to the correct dynamical regime. Different parameters influence the dynamics of the reservoir, such as the size of the reservoir, the type of nonlinearity and the topology. Optimal values for all these parameters depend on the specific task which needs to be solved.

The number of publications in Reservoir Computing has grown tremendously over the past few years. The popularity of the approach lies in its simplicity and good performance on a variety of tasks, such as speech recognition [27], detection of epileptic seizures [39], robot localization [40], time-series prediction [41, 42] and so on. Most commonly, the reservoir is of the ESN type, i.e., it uses sigmoidal analog neurons to simulate the RNN.

2.3.1 Mathematical description of the reservoir

In this section we give a more rigorous mathematical description of a RC system. We depict the system in Figure 2.5.

The states of a reservoir of size N at timestep k are then given by $\mathbf{x}[k]$ ⁷, $k \in \mathbb{N}$. $\mathbf{x}[k]$ is a column vector with dimensions $(N, 1)$. The next state of the reservoir is calculated using the following update equation:

$$\mathbf{x}[k+1] = (1 - \eta)\mathbf{x}[k] + \eta\mathbf{f}(\mathbf{W}_{in}\mathbf{u}[k] + \mathbf{W}_{res}\mathbf{x}[k]), \quad (2.4)$$

where \mathbf{W}_{in} (N, K) is the weight matrix to feed K inputs $\mathbf{u}[k]$ to the reservoir, \mathbf{W}_{res} (N, N) is the connection matrix of the reservoir with N neurons and η is the leak rate of the neurons.

Typically, a reservoir is simulated in discrete time. Although in photonics we are inherently making use of a continuous-time system, for this introduction we will limit ourselves to a discrete-time system⁸.

f is the nonlinear activation function. As we mentioned before, in most applications we will use a hyperbolic tangent function:

$$\tanh(x) = \frac{\exp(x) - \exp(-x)}{\exp(x) + \exp(-x)}. \quad (2.5)$$

An important property of a RNN is its spectral radius. The spectral radius is defined as the absolute value of the largest eigenvalue of the system's Jacobian at its maximal gain state. For a hyperbolic tangent reservoir where each neuron is described by $\mathbf{f}(\mathbf{x}) = \tanh(\mathbf{x})$, all neurons have a maximal gain of one and the spectral radius can be simplified to:

$$SR = \max(|\text{eig}(\mathbf{W}_{res})|). \quad (2.6)$$

The spectral radius gives an indication of the stability of the network. In a nonlinear reservoir however, the actual gain is dependent on the current states of the neurons. If this value on average exceeds 1, then instability can occur. In most cases, a value close to 1 is desirable. Intuitively, the reservoir should react with strong dynamics to input signals from different classes so that the problem becomes linearly separable. This enables the linear readout to more easily perform the classification. For this reason, the spectral radius should be chosen large, because then it is more sensitive to the input. On the other hand, if the reservoir is too dynamic (and even chaotic or unstable), two slightly different inputs can lead to two completely different trajectories of the system, and it

⁷In the remainder of this dissertation, we will use square brackets to denote a discrete-time dependency, such as $\mathbf{x}[k]$, and round brackets to denote a continuous-time dependency, such as $\mathbf{x}(t)$.

⁸The same theories and learning methods can be applied to continuous-time systems, if we sample the system at fixed time intervals, i.e., $\mathbf{x}[k] = \mathbf{x}(\Delta t k)$. Given the typical timescales in photonics, Δt is typically between ps (10^{-12} s) and ns (10^{-9} s).

becomes impossible for the readout to extract useful information from the system. This optimal regime is usually referred to as the edge of chaos (or rather, the edge of stability). A rigorous proof on why this is beneficial is out of the scope of this introduction, but can be found in chapter 2 of the PhD thesis of D. Verstraeten, and in [43, 44].

Usually, the reservoir is constructed as follows (there are many variations on this scheme, but for simplicity we only show the most common way to construct the reservoir):

1. $\mathbf{W}_{res}(N, N)$: The weights are drawn from a continuous random distribution (e.g., a gaussian distribution with zero mean and unit standard deviation) or discrete set (e.g., $\{-1, 1\}$), and globally scaled to reach a certain spectral radius (see equation (2.6)). Usually, a good starting value is a value close to, but smaller than 1. Often, only a fraction (usually 10%) of the elements of \mathbf{W}_{res} are non-zero. This determines the sparsity of the topology. For typical benchmark tasks, around $N = 1000$ neurons are used.
2. $\mathbf{W}_{in}(N, K)$: The input weights are constructed in the same way. They are globally scaled with the *input scale factor*, and can be either sparse or fully connected.

2.3.2 Training the reservoir

There are two different approaches to train a reservoir: off-line and on-line learning.

Off-line learning In this technique, all input signals are fed to the network and the dynamical responses of the network $\mathbf{x}[k]$ are recorded. Afterward, the output weights \mathbf{W}_{out} are trained by using all this data and the desired outputs of the system. The speech task which we will investigate in chapter 5 has been trained off-line.

On-line learning In on-line learning, the learning is done incrementally. The output weights are modified immediately, or soon after a new sample is acquired. In chapter 6, we will train a network to generate periodic patterns using an on-line learning rule.

Since we will use both methods in this dissertation, we will elaborate on the training procedures in the follow two sections.

2.3.3 Off-line training

Consider a dataset which consists of a number of examples, each of them being a number of samples long. The dataset is first split into a train set I and a test-

ing set J . For the train set, $T_{train,i}$ is the length of the i -th example. Summing them gives the total number of samples $D_{train} = \sum_i T_{train,i}$, for $i \in I$. Similarly, $D_{test} = \sum_j T_{test,j}$, for $j \in J$. In a first phase we feed all training samples sequentially to the reservoir, and collect the states $\mathbf{x}[k]$ in a large state matrix \mathbf{A} (D_{train}, N). Furthermore, \mathbf{B} (D_{train}, L) is a matrix with the desired L outputs. The problem we want to solve is the following: find proper output weights \mathbf{W}_{out} (N, L), corresponding to the linear readout layer, such that

$$\mathbf{A} \cdot \mathbf{W}_{out} = \mathbf{B}. \quad (2.7)$$

We assume this system is overdetermined, which is normally the case as the number of neurons is fixed and usually we have many samples to feed to the neural network. To solve this problem, we perform a least squares linear regression which minimizes the difference (calculated as a sum of squares) between the desired output and the output of the reservoir:

$$\mathbf{W}_{out} = \min_{\mathbf{W}} \|\mathbf{A} \cdot \mathbf{W} - \mathbf{B}\|^2. \quad (2.8)$$

To solve this problem we can use the pseudo-inverse of the matrix \mathbf{A} , also called the Moore-Penrose generalized matrix inverse [45]:

$$\mathbf{A}^\dagger = (\mathbf{A}^H \mathbf{A})^{-1} \mathbf{A}^H \quad (2.9)$$

Where we use \mathbf{A}^H to denote the conjugate transpose of the matrix \mathbf{A} . The solution for \mathbf{W}_{out} is then given by:

$$\mathbf{W}_{out} = \mathbf{A}^\dagger \mathbf{B} = (\mathbf{A}^H \mathbf{A})^{-1} \mathbf{A}^H \mathbf{B} \quad (2.10)$$

To our knowledge, all RC systems that have been studied before have a real-valued readout layer. This also holds for the photonic simulations that were performed in this dissertation: although optical signals are complex-valued, we first transform them to real-valued signals, either by using the magnitude of the signal or by splitting it into a real and an imaginary part. For the remainder of this dissertation we will assume that \mathbf{A} is real, so instead of using the conjugate transpose of \mathbf{A} we can use the regular transpose: \mathbf{A}^T .

After calculating the output weights, we can test the system by feeding the test data into the reservoir. If the response of the system on the test inputs is given by \mathbf{A}' , then the output of the system is given by

$$\mathbf{y} = \mathbf{W}_{out} \mathbf{A}'. \quad (2.11)$$

The final performance of the RC system is then calculated by using some metric involving the desired output of the testing set \mathbf{B}' and the actual output \mathbf{y} (for example, using the mean square error).

In the previous method, we have considered a train and a testing set, which is usually the case for classification tasks. Another class of tasks exists in which we want to predict the next time sample. This is called one-step-ahead prediction, and can be solved in the same way. The input is a time-series, and the desired output is the same time-series but shifted by one timestep. This method can also be used in signal generation tasks where it is called teacher-forcing. In this case, a feedback loop is created after the training phase, replacing the input by the trained reservoir output. Now the system has to autonomously generate its next timestep. This is particularly relevant for robotics applications. The network can have difficulties generalizing and can be very sensitive to noise after training, but regularization and pruning can improve these results [46]. In the next section, we discuss regularization in more detail.

2.3.3.1 Regularization

In solving equation (2.10), a big issue is the problem of overfitting. Overfitting means that a model fails to capture the underlying properties of the data, and usually occurs when the system is learning noise rather than trying to generalize the problem. Overfitted models will typically perform very well on the training set but very badly on the testing set. Consider, as an example, two young students that are learning the multiplication tables with numbers above 10. The first student is told to learn all combinations by heart. The second student is taught to actually multiply the numbers and understand how $15 \cdot 6$ is actually just $10 \cdot 6 + 5 \cdot 6$. The first student, roughly speaking, is overfitting and will not be able to multiply a combination of two numbers it hasn't seen before. The second student however, learns how to perform a general multiplication, and is able to understand the underlying properties of the data⁹. The same happens to the reservoir: if the system has too many parameters to be trained, there is a risk that these parameters will capture the noise rather than the underlying properties of the data. The solution to this problem is to use regularization. This is done by adding a penalty term (scaled by γ) for large output weights to the minimization problem:

$$\mathbf{W}_{opt,\gamma} = \min_{\mathbf{W}} (\|\mathbf{A} \cdot \mathbf{W} - \mathbf{B}\|^2 + \gamma \|\mathbf{W}\|^2). \quad (2.12)$$

As norm, we have used the standard Euclidean norm¹⁰, given by the sum of squares of each item:

$$\|\mathbf{W}\|^2 = \sum_{n=0}^{n=N-1} \sum_{l=0}^{l=L-1} \mathbf{W}_{n,l}. \quad (2.13)$$

⁹Arguably, there's an optimum where people use a combination of both remembering by heart, and performing the calculation, in order to score high on a multiplication test.

¹⁰Other norms can be used too, giving rise to different properties of the regularization.

This type of regularization (with Euclidean norm) is called ridge regression [47] or Tikhonov regression [48], where γ is the ridge regression parameter. The optimal output weights are found to be:

$$\mathbf{W}_{out} = (\mathbf{A}^T \mathbf{A} - \gamma \mathbf{I})^{-1} \mathbf{A}^T \mathbf{B} \quad (2.14)$$

Another way to perform regularization is by adding noise to the recorded state values of the dynamical system before training [47]. However, adding noise is inherently non-deterministic (which means reproducing results is more difficult), and the system has to be optimized for different noise levels. In contrast, the ridge regression parameter can be optimized with an algorithm that is independent of the size of the dataset, which makes it computationally more efficient for large datasets [49]. So in practice, the ridge regression method is usually preferred.

2.3.3.2 Cross-validation

Correctly measuring the performance of a reservoir consists of several steps. First, the dataset is split into a training set, a validation set, and a testing set. Using the training set and validation set, one can optimize the parameters of a reservoir (including the spectral radius, input scaling, but also the regression parameter). With these optimal parameters, the performance is again measured using the testing set, which contains unseen data.

If the number of samples is small, the choice of training and validation set can influence the measured performance. For example, a class that needs to be classified might be missed in the chosen training set, leading to misinterpretation of the results. To this end we will simulate the reservoir for different combinations of training and validation sets. This is called cross-validation. Furthermore, in this process, we will look for the optimal ridge regression parameter γ for each choice of training and validation samples.

It is easier to illustrate this process with an example (in fact, we will use this method for the isolated digit recognition task in chapter 5). Consider there are 500 spoken digits. We combine the first 100 digits in set 1, the next 100 digits in set 2, and so on. We now have 5 sets. To perform cross-validation, we will first consider the training set 1,2,3,4, and test with the spoken digits from set 5. This is called one fold. Next we will train using the samples from sets 1,2,3,5, and use set 4 to test. This is the second fold. And so on. In this example, we have $F = 5$ folds, where each set has been used once for testing, while the other sets are used for training. For each fold f , we have trained the system for several γ , logarithmically spaced to cover a broad range, and selected the best γ_f for this fold.

Ideally, for very accurate results, there should be as many folds F as there

are spoken digits (this is called leave one out cross-validation). On the other hand, the computational resources needed increase with larger F . A trade-off between this computational cost and selection accuracy has to be made, and for the speech task we chose $F = 5$.

In this dissertation, we are mainly focussed on performance trends that we can discover as a function of the large amount of parameters. Furthermore there are only a relatively few samples (500 samples). For this reason we decide, as in K. T. Vandoorne's doctoral thesis, not to use a testing set, so that we have more samples for the parameter optimization. This means that the absolute WER has no actual meaning (indeed, it would be unfair to re-use samples during testing¹¹). The advantage is that parameter trends will be slightly more accurate due to having more samples in the training and validation set.

2.3.3.3 Unbalanced datasets and Fisher relabeling

Suppose a classification task has to distinguish between two classes A and B. The output of the linear classifier should be 1 if the system recognizes class A, and -1 if it recognizes class B. This can be done by either performing a least squares regression (which minimizes the quadratic error between the projection of each point onto a hyperplane), or by using the Fisher linear discriminant¹², which creates a linear hyperplane by minimizing the within-class variance, while at the same time maximizing the between-class variance.

There is a problem with least squares regression: if there are much fewer examples of class A (n_1 examples) than there are of class B (n_2 examples), the hyperplane constructed by the linear readout layer typically shifts towards the class with more examples (in this case class B), and it can happen that the two classes are not separated well by the hyperplane anymore. This means that the reservoir will be less capable of generalizing when it sees new data. This is illustrated in Figure 2.6, where the dashed line represents the separating hyperplane which is likely to yield more errors due to the unbalanced dataset.

To compensate for this unbalance, we can relabel the outputs of the classifier. Instead of using 1, -1, we will use $\frac{n_1+n_2}{n_1}$, $-\frac{n_1+n_2}{n_2}$. The new labels reflect the unbalance and counteract the shifting of the hyperplane. In [50] it is shown that this type of relabeling is equivalent to using the Fisher linear discriminant (see also [51]), hence the name *Fisher relabeling*.

The same issue will occur in the isolated digit recognition task. In this task, one wishes to make a distinction between 10 spoken digits. This means we will use ten classifiers, one for each spoken digit. Each classifier should return a

¹¹As it would be unfair to have the examination questions before going to the actual exam.

¹²Often the Fisher linear discriminant is interchanged with Linear Discriminant Analysis (LDA). The latter however uses a few more assumptions, such as normal distributed classes and equal class covariance.

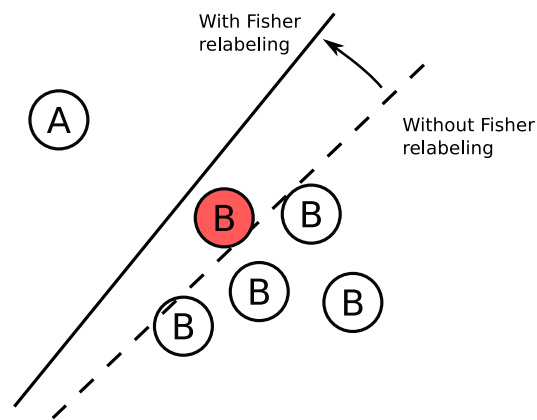


Figure 2.6: In an unbalanced dataset, the linear separation plane can shift towards the class with more samples. In this illustration two classes, A and B are used. Because there are more samples of class B than there are of class A, the separating plane (dashed lines) will be skewed towards class B. In this case, the example of class B marked in red is classified wrongly. By relabeling the weights (in this case, increasing the strength of label 'A', and decreasing the strength of label 'B'), the resulting separation plane (full line) is more accurate, leading to a correct classification of the red 'B'. This technique is also called Fisher relabeling.

positive or negative response based on whether the corresponding digit was applied on the input of the reservoir (positive) or not (negative). Without re-labeling, there is a big unbalance in the dataset: when each digit has the same number of examples, the classifier for the spoken digit '1' only has 1/10th of the examples that return a positive response. Using the Fisher relabeling technique we can compensate for this unbalance.

2.3.4 On-line learning

The disadvantage of off-line learning is that the learning is performed in batches, in which all data has to be known before training. In on-line learning, the output weights are changed during training and new information can be added to the network during the learning process. For example in stock market prediction, as soon as a prediction is made, the actual stock market conditions are available and the system can correct for errors it made in predicting the current condition.

On-line learning is used often for training systems that predict the behavior of a nonlinear dynamical system such as the Mackey-Glass system [52], the Nonlinear Autoregressive Moving Average (NARMA) task and robot locomotion.

There are several ways to perform on-line learning. In the introduction we have talked about BackPropagation DeCorrelation (BPDC) [36]. Another very popular way to modify the weights is by using the Recursive Least Squares (RLS) algorithm, which is a special case of the discrete Kalman filter [53, 54]. The RLS rule is used extensively in the last chapter of this thesis, so we will give a brief mathematical description of the method here.

The RLS method wishes to minimize the difference between the output $\mathbf{y}[k]$ of a system and a desired function $\mathbf{s}[k]$ by recursively modifying the readout weights. We define the difference between the output neuron and the desired function as follows:

$$\mathbf{e}_-[k] = \mathbf{W}_{out}[k-1]\mathbf{x}[k] - \mathbf{s}[k] \quad (2.15)$$

In which the minus subscript is used to denote the error prior to the weight update at time k . The next step in the process is to update the weights in such a way that it reduces the magnitude of $\mathbf{e}_-[k]$. The error after the weight update equals:

$$\mathbf{e}_+[k] = \mathbf{W}_{out}[k]\mathbf{x}[k] - \mathbf{s}[k] = \mathbf{y}[k] - \mathbf{s}[k] \quad (2.16)$$

All weight modification schemes have a simple goal: reduce errors during training ($|\mathbf{e}_+(t)| < |\mathbf{e}_-(t)|$) and converge to a stable solution where the weights do not change anymore ($\mathbf{e}_+(t)/\mathbf{e}_-(t) \rightarrow 1$).

In the RLS modification scheme, the output weights \mathbf{W}_{out} are modified each timestep according to the following rule:

$$\mathbf{W}_{out}[k] = \mathbf{W}_{out}[k-1] - \mathbf{e}_-[k]\mathbf{P}[k]\mathbf{r}[k]. \quad (2.17)$$

Here, we used $\mathbf{r}_i = \tanh(\mathbf{x}_i)$. The outputs are sometimes referred to as firing rates, which clearly refers to the firing-rate descriptions of a neuron¹³.

In a nanophotonic network, there is no equivalent of the firing rate. Instead, to modify the weights, we will use the following equation:

$$\mathbf{W}_{out}[k] = \mathbf{W}_{out}[k-1] - \mathbf{e}_-[k]\mathbf{P}[k]\mathbf{s}[k], \quad (2.18)$$

where $\mathbf{s}[k]$ can be either the complex amplitudes $\mathbf{a}[k]$ corresponding to the mode of a photonic crystal cavity (see section 3.2.1), or we can use path splitters to 'tap' an amount of power from the network to a detector and use that for the readout.

The $\mathbf{P}[k]$ matrix is a $(N \times N)$ matrix that is updated at the same time as the weights according to the following rule:

$$\mathbf{P}[k] = \mathbf{P}[k-1] - \frac{\mathbf{P}[k-1]\mathbf{x}[k]\mathbf{x}^T[k]\mathbf{P}[k-1]}{1 + \mathbf{r}^T\mathbf{P}[k-1]\mathbf{r}[k]}. \quad (2.19)$$

Here \mathbf{P} is a running estimate of the inverse of the correlation matrix of the network rates \mathbf{r} plus a regularization term, i.e., $(\sum_k \mathbf{r}[k]\mathbf{r}^T[k] + \alpha\mathbf{I})$ [55]. The initial value for \mathbf{P} is given by:

$$\mathbf{P}[0] = \frac{\mathbf{I}}{\alpha}, \quad (2.20)$$

in which α acts as a learning rate. The convergence properties of this system are explained in more detail in D. Sussillo's paper [56]. In summary, α should be significantly smaller than the number of neurons N , but not too small, because then the weights can change so rapidly that the algorithm becomes unstable. When α becomes too large, the training converges more slowly and training can even fail. Typical values of α (for a network of 1000 neurons) range between 1 and 100.

A much simpler rule exists, that does not use a full matrix \mathbf{P} but a single *scalar* time-dependent learning rate $\eta(t)$. The weight modification rule is given by:

$$\mathbf{W}_{out}[k] = \mathbf{W}_{out}[k-1] - \eta[k]\mathbf{e}_-[k]\mathbf{r}[k], \quad (2.21)$$

Although this basic *delta* rule cannot generate the same complexity of signals and converges more slowly than the RLS rule, it is much easier to implement, from a hardware point of view, and more biologically plausible.

¹³The analog tanh neuron is an approximation of a spiking neuron, where the output of the neuron (given by equation (2.3)) represents the firing rate. Conceptually, this firing rate is proportional to the repetition rate of spikes that are fired by the spiking neuron.

2.3.4.1 Generating periodic patterns using FORCE

In 2009, a new technique called FORCE was proposed by D. Sussillo and L.F. Abbott [56]. In their paper, the authors describe a technique which is used to modify the chaotic spontaneous activity of a RNN into a wide variety of desired patterns, such as the motor output for steering a robot leg. Initially, the authors follow the ESN approach, where only a layer of output neurons is trained, but later they use different architectures in which the internal weights of the RNN are also trained, which is, from a biological point of view, more realistic. In this dissertation we restrict ourselves to the ESN architecture, because it is easier to implement, especially when designing a hardware reservoir, and computationally less demanding.

In the case of a signal generation task, it forms an alternative to the off-line teacher forced learning method.

The output neuron generates the periodic pattern which we want to train, which is fed back to the input neuron, as shown in Figure 2.7. There are several important differences with previous on-line learning approaches. First, in this rule, it is no longer the desired signal $\mathbf{s}[k + 1]$ which is fed to the input during training (one-step-ahead prediction as in teacher forcing), but rather the actual network output $\mathbf{y}[k + 1]$. The learning rule (either RLS or the delta rule), which keeps the error small, allows the system to sample instabilities in the RNN and stabilize them. This difference, although numerically small, has an extremely significant implication for the network stability. Because the method requires the (first-order) error to be kept small, the authors call this method First-Order Reduced and Corrected Error (FORCE).

Second, the spectral radius of the network (see equation (2.6)) is chosen larger than one, which means that the network is initially unstable. As we have seen before, a larger spectral radius means the network can respond with richer dynamics, which can lead to better separation of trajectories in the high-dimensional phase space. If it is chosen too high, the system becomes unstable and cannot give sensible information of the inputs anymore. However, it has been shown in [57, 58] that the correct type of input (here: feeding back the neuron output after the weight change) can induce a transition between chaotic and nonchaotic states. Although the systems described in [56] are based on leaky hyperbolic tangent neurons, we will show in this work that the FORCE technique also works for nonlinear nanophotonic cavities, which can also exhibit chaos when excited in certain regimes before the training phase.

Third, as opposed to other weight modification methods where the errors $\mathbf{e}_+(t)$ and $\mathbf{e}_-(t)$ can be quite large, the weight changes in FORCE learning are kept small during the simulation. This is done by making one large jump in the output weights in the beginning of the simulation, in order to come close to the desired signal, and then keeping the error small using an appropriate learning

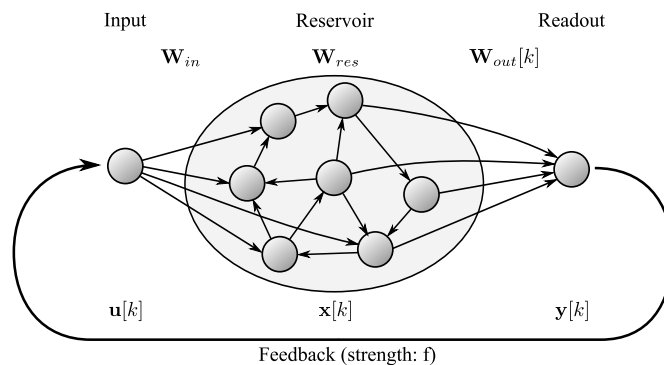


Figure 2.7: An RNN network with feedback. Note that the output weights $\mathbf{W}_{out}[k]$ are now time dependent. The output is fed back to the input. The purpose of this network is to autonomously generate periodic patterns after \mathbf{W}_{out} has been trained.

rule, such as the RLS rule.

Thanks to these three properties, the FORCE learning rule is very robust (it converges easily) and stable (meaning that after a very long time the system is still able to generate the same pattern).

Chapter 6 is fully devoted to the training of nanophotonic reservoirs using this novel technique. We will demonstrate that it is possible to train a nanophotonic reservoir that autonomously generates periodic patterns.

2.4 Software framework

All reservoir simulations performed in this thesis have been performed using the the OrGanic Environment for Reservoir computing (OGER) [59]. OGER is used to set up, simulate and post-process RC experiments. OGER is publicly available¹⁴, is written in the Python scripting language, and it extends the MDP toolbox [60], by providing extra functionality in the context of machine learning in general and reservoir computing in particular. It contains different standard reservoirs, datasets, tasks and and it contains most training mechanisms that were described in this chapter.

References

- [1] D. Verstraeten. *Reservoir Computing: Computation with Dynamical Systems*. Phd, Ghent University, 2009.

¹⁴<http://reservoir-computing.org/organic/engine>

- [2] Alan M. Turing. *On computable numbers, with an application to the Entscheidungsproblem*. Proceedings of the London Mathematical Society, 42(2):230–265, 1936.
- [3] C. Eliasmith, T.C. Stewart, X. Choo, T. Bekolay, T. DeWolf, C. Tang, and D. Rasmussen. *A large-scale model of the functioning brain*. science, 338(6111):1202–1205, 2012.
- [4] Vladimir N. Vapnik. *The nature of statistical learning theory*. Springer-Verlag New York, Inc., New York, NY, USA, 1995.
- [5] J.L. Rumelhart, D.E.; McClelland. *Parallel Distributed Processing*, chapter Learning internal representations by error propagation. MIT Press, Cambridge, MA, 1986.
- [6] T. Kohonen, MR Schroeder, TS Huang, and S.O. Maps. *Springer-Verlag New York. Inc.*, Secaucus, NJ, 2001.
- [7] J.A. Hartigan. *Clustering algorithms*. John Wiley & Sons, Inc., 1975.
- [8] N.N. Schraudolph, P. Dayan, T.J. Sejnowski, et al. *Temporal difference learning of position evaluation in the game of Go*. Advances in Neural Information Processing Systems, pages 817–817, 1994.
- [9] G. Tesauro. *Temporal difference learning and TD-Gammon*. Communications of the ACM, 38(3):58–68, 1995.
- [10] B. DasGupta and G. Schnitger. *The power of approximating: a comparison of activation functions*. Advances in neural information processing systems, 5:615–622, 1993.
- [11] Arthur E. Bryson and Yu-Chi Ho. *Applied optimal control; optimization, estimation, and control*. Blaisdell Pub. Co. Waltham, Mass., 1969.
- [12] Paul J. Werbos. *Beyond Regression: New Tools for Prediction and Analysis in the Behavioral Sciences*. PhD thesis, Harvard University, 1974.
- [13] W. Maass and C.M. Bishop. *Pulsed neural networks*. MIT press, 2001.
- [14] Benjamin Schrauwen. *Towards applicable spiking neural networks*. PhD thesis, Universiteit Gent, 2008.
- [15] Wolfgang Maass. *Networks of spiking neurons: The third generation of neural network models*. Neural Networks, 10(9):1659 – 1671, 1997.
- [16] A. Waibel, T. Hanazawa, G. Hinton, K. Shikano, and K.J. Lang. *Phoneme recognition using time-delay neural networks*. Acoustics, Speech and Signal Processing, IEEE Transactions on, 37(3):328–339, 1989.

- [17] J.A.K. Suykens, J.P.L. Vandewalle, and B.L. de Moor. *Artificial neural networks for modelling and control of non-linear systems*. Springer, 1995.
- [18] A.J. Robinson. *An application of recurrent nets to phone probability estimation*. Neural Networks, IEEE Transactions on, 5(2):298–305, 1994.
- [19] A. Graves, D. Eck, N. Beringer, and J. Schmidhuber. *Biologically plausible speech recognition with LSTM neural nets*. Biologically Inspired Approaches to Advanced Information Technology, pages 127–136, 2004.
- [20] A. Graves and J. Schmidhuber. *Offline handwriting recognition with multidimensional recurrent neural networks*. Advances in Neural Information Processing Systems, 21:545–552, 2009.
- [21] B. Hammer and J. J. Steil. *Perspectives on learning with recurrent neural networks*. In Proceedings of the 10th European Symposium on Artificial Neural Networks (ESANN), Bruges, Belgium, April 2002.
- [22] P. Werbos. *Beyond regression: New tools for prediction and analysis in the behavioral sciences*. 1974.
- [23] P.J. Werbos. *Backpropagation through time: what it does and how to do it*. Proceedings of the IEEE, 78(10):1550–1560, 1990.
- [24] R.J. Williams and D. Zipser. *A learning algorithm for continually running fully recurrent neural networks*. Neural computation, 1(2):270–280, 1989.
- [25] G.V. Puskorius and L.A. Feldkamp. *Neurocontrol of nonlinear dynamical systems with Kalman filter trained recurrent networks*. Neural Networks, IEEE Transactions on, 5(2):279–297, 1994.
- [26] D.V. Prokhorov. *Training recurrent neurocontrollers for real-time applications*. Neural Networks, IEEE Transactions on, 18(4):1003–1015, 2007.
- [27] Benjamin Schrauwen, Michiel D’Haene, David Verstraeten, and Jan Van Campenhout. *Compact hardware liquid state machines on FPGA for real-time speech recognition*. Neural Networks, page 2008, 2008.
- [28] Felix Schürmann, Karlheinz Meier, and Johannes Schemmel. *Edge of chaos computation in mixed-mode vlsi - a hard liquid*. In In Proc. of NIPS. MIT Press, 2005.
- [29] L. Appeltant, M.C. Soriano, G. Van der Sande, J. Danckaert, S. Massar, J. Dambre, B. Schrauwen, C.R. Mirasso, and I. Fischer. *Information processing using a single dynamical node as complex system*. Nature Communications, 2011.

- [30] Kristof Vandoorne, Joni Dambre, David Verstraeten, Benjamin Schrauwen, and Peter Bienstman. *Parallel reservoir computing using optical amplifiers*. IEEE Transactions On Neural Networks, 22(9):1469–1481, 2011.
- [31] Mantas Lukoševičius, Herbert Jaeger, and Benjamin Schrauwen. *Reservoir Computing Trends*. KI - Künstliche Intelligenz, accepted 2012.
- [32] P.F. Dominey. *Complex sensory-motor sequence learning based on recurrent state representation and reinforcement learning*. Biological Cybernetics, 73(3):265–274, 1995.
- [33] D.V. Buonomano, M.M. Merzenich, et al. *Temporal information transformed into a spatial code by a neural network with realistic properties*. SCIENCE-NEW YORK THEN WASHINGTON-, pages 1028–1028, 1995.
- [34] H. Jaeger. *The echo state approach to analysing and training recurrent neural networks*. Technical report GMD report, 148, 2001.
- [35] W. Maass, T. Natschläger, and H. Markram. *Real-time computing without stable states: A new framework for neural computation based on perturbations*. Neural computation, 14(11):2531–2560, 2002.
- [36] J.J. Steil. *Backpropagation-Decorrelation: Online recurrent learning with $O(N)$ complexity*. In Neural Networks, 2004. Proceedings. 2004 IEEE International Joint Conference on, volume 2, pages 843–848. IEEE, 2004.
- [37] D. Verstraeten, B. Schrauwen, and D. Stroobandt. *Isolated word recognition using a liquid state machine*. In Proceedings of the 13th European Symposium on Artificial Neural Networks (ESANN), pages 435–440, 2005.
- [38] P. Joshi and W. Maass. *Movement generation with circuits of spiking neurons*. Neural Computation, 17(8):1715–1738, 2005.
- [39] Pieter Buteneers, David Verstraeten, Pieter van Mierlo, Tine Wyckhuys, Dirk Stroobandt, Robrecht Raedt, Hans Hallez, and Benjamin Schrauwen. *Automatic detection of epileptic seizures on the intra-cranial electroencephalogram of rats using reservoir computing*. Artificial Intelligence In Medicine, 53(3):215–223, 2011.
- [40] Eric Antonelo, Benjamin Schrauwen, and Dirk Stroobandt. *Event detection and localization for small mobile robots using reservoir computing*. Neural Networks, 21(6):862–871, 2008.
- [41] Herbert Jaeger and Harald Haas. *Harnessing Nonlinearity: Predicting Chaotic Systems and Saving Energy in Wireless Communication*. Science, 304(5667):78–80, 2004.

- [42] F. Wyffels and B. Schrauwen. *A comparative study of Reservoir Computing strategies for monthly time series prediction*. *Neurocomputing*, 73(10–12):1958–1964, 2010.
- [43] C.G. Langton. *Computation at the edge of chaos: Phase transitions and emergent computation*. *Physica D: Nonlinear Phenomena*, 42(1):12–37, 1990.
- [44] R. Legenstein, W. Maass, et al. *Edge of chaos and prediction of computational performance for neural circuit models*. *Neural Networks*, 20(3):323–334, 2007.
- [45] R. Penrose. *A generalized inverse for matrices*. In *Proc. Cambridge Philos. Soc*, volume 51, page C655. Cambridge Univ Press, 1955.
- [46] X. Dutoit, B. Schrauwen, J. Van Campenhout, D. Stroobandt, H. Van Brussel, and M. Nuttin. *Pruning and regularization in reservoir computing*. *Neurocomputing*, 72(7-9):1534–1546, 2009.
- [47] Francis Wyffels, Benjamin Schrauwen, and Dirk Stroobandt. *Stable Output Feedback in Reservoir Computing Using Ridge Regression*. In V. Kurkova, R. Neruda, and J. Koutnik, editors, *Proceedings of the 18th International Conference on Artificial Neural Networks*, pages 808–817, Prague, 9 2008. Springer.
- [48] A. Tikhonov and V. Y. Arsenin. *Solutions of Ill-Posed Problems*. 1977.
- [49] Pieter Buteneers, Ken Caluwaerts, David Verstraeten, and Benjamin Schrauwen. *Optimization of the Regularization Parameter and Feature Selection for Ridge Regression: Fast Algorithms for Large Datasets*. *Neurocomputing*, submitted (2011).
- [50] R. O. Duda, P. E. Hart, and D. G. Stork. *Pattern Classification (2nd edition)*. John Wiley and Sons, Inc., New York, USA, 2001.
- [51] R.A. Fisher. *The use of multiple measurements in taxonomic problems*. *Annals of Human Genetics*, 7(2):179–188, 1936.
- [52] J.J. Steil et al. *Online reservoir adaptation by intrinsic plasticity for backpropagation-decorrelation and echo state learning*. *Neural Networks*, 20(3):353–364, 2007.
- [53] S. Singhal and L. Wu. *Training multilayer perceptrons with the extended Kalman algorithm*. In *Advances in neural information processing systems 1*, pages 133–140. Morgan Kaufmann Publishers Inc., 1989.

-
- [54] A.H. Sayed and T. Kailath. *A state-space approach to adaptive RLS filtering*. Signal Processing Magazine, IEEE, 11(3):18–60, 1994.
- [55] S. Haykin. *Adaptive Filter Theory*. Upper Saddle River, NJ: Prentice hall), 2002.
- [56] David Sussillo and L. F. Abbott. *Generating Coherent Patterns of Activity from Chaotic Neural Networks*. Neuron, 63(4):544–557, AUG 27 2009.
- [57] L. Molgedey, J. Schuchhardt, and HG Schuster. *Suppressing chaos in neural networks by noise*. Physical review letters, 69(26):3717–3719, 1992.
- [58] N. Bertschinger and T. Natschläger. *Real-time computation at the edge of chaos in recurrent neural networks*. Neural Computation, 16(7):1413–1436, 2004.
- [59] David Verstraeten, Benjamin Schrauwen, Sander Dieleman, Philemon Brakel, Pieter Buteneers, and Dejan Pecevski. *Oger: Modular Learning Architectures For Large-Scale Sequential Processing*. Journal of Machine Learning Research (submitted), 2011.
- [60] Tiziano Zito, Niko Wilbert, Laurenz Wiskott, and Pietro Berkes. *Modular toolkit for Data Processing (MDP): a Python data processing framework*. Frontiers in Neuroinformatics, 2(00008), 2009.

3

Photonic Crystal Cavities

Photonic crystals allow us to fully control the propagation of light in a dielectric medium. Because of this, very compact structures can be made, and moreover, light can be trapped in a very small volume (photonic crystal cavities), locally increasing the intensity of the field. This allows us to take advantage of the non-linear properties of the material, which are useful for many reservoir tasks. This thesis is focused on performing reservoir computations with photonic crystal cavities, and thus it is very important to understand the physics behind them. This is explained in section 3.1.

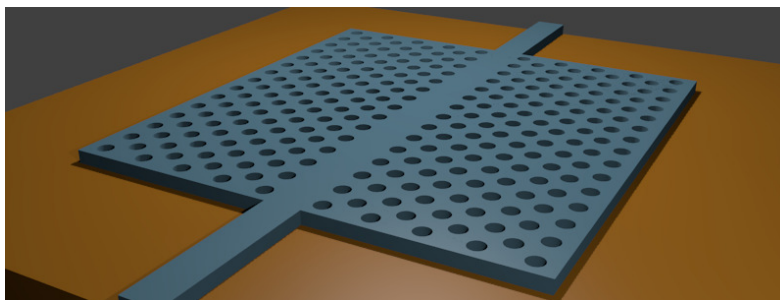


Figure 3.1: A 2D photonic crystal cavity with a line defect (W1 defect).

During this research we have performed a lot of simulations based on these photonic crystal cavities¹. For example, full-wave electromagnetic simulations such as a Finite Difference Time Domain simulations (FDTD) are very accurate, but consume a lot of time and become impractical for 100s of elements. Because of this, we seek descriptive models that need only a fraction of the computation power (both in terms of speed and memory). In section 3.2.1 we explain the Coupled Mode Theory (CMT), a simplified analytical model that allows us to focus on the dynamics of these cavities rather than performing brute-force simulations in FDTD. To check whether the CMT is accurate enough, we perform a series of FDTD simulations and compare the results of these simulations with the simplified model in 3.2.2. We find that the correspondence between the two methods is very good, which means that from now on we can use the simplified model for simulating large circuits.

Although the emphasis of this thesis is on theoretical results, we have already designed and measured photonic crystal cavities (section 3.3). We can use these measurement results to better match the simulation parameters to realistic values and, more importantly, they give us an idea of how close we are to fabricating a full nanophotonic reservoir using the standard SOI technology (as we have seen in section 1.2).

3.1 Photonic crystals: principle

The book *Molding the Flow of Light* [1] is an excellent reference for all theory on photonic crystals. Although a full description is not possible here, we will summarize the most important concepts in the next section. We briefly describe why we chose to perform 2D simulations in section 3.1.2 and explain how nonlinearities are relevant to these cavities in section 3.1.3.

3.1.1 Periodicity and band gaps

The most important thing about photonic crystals is that they are periodic, as can be seen in Figure 3.1. The repeating pattern is called the crystal lattice. Light can be represented as a wave, and waves that meet certain criteria can travel through a periodic potential without scattering (apart from impurities and defects in the lattice). These are the so-called guided Bloch modes. Additionally, a lattice can have a range of energies and directions in which light cannot propagate. This is called a band gap. If, for some frequency range, a photonic crystal prohibits the propagation of electromagnetic waves of any polarization traveling in any direction, we say that the crystal has a complete photonic band gap.

¹In the next chapter we give a more broad overview of possible modeling methods (see section 4.1). We only discuss Finite Difference Time Domain and Coupled Mode Theory here.

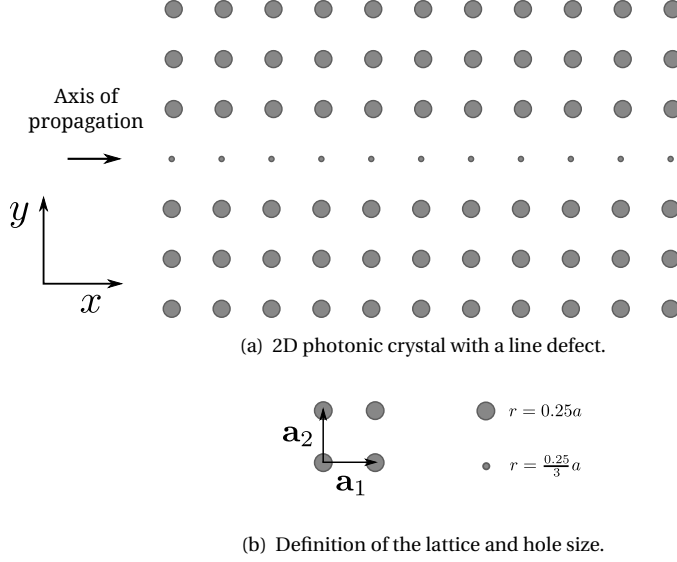


Figure 3.2: A 2D photonic crystal with rectangular lattice. The rods have a high dielectric constant, the surrounding has a low dielectric constant. Light propagates in the horizontal direction through a line defect (also called a W1 defect). The lattice is defined by the two vectors \mathbf{a}_1 and \mathbf{a}_2 , both have magnitude a . The rods have a radius defined by $r = 0.25a$ and the defect rods have a radius $r = \frac{0.25}{3}a$. These parameters will be used later on for our simulations.

In the rest of the study we restrict ourselves to 2D photonic crystals, as this is the structure we investigated mostly in this thesis (in section 3.1.2 we briefly explain what happens when adding a third dimension).

In Figure 3.2(a) we show a two-dimensional photonic crystal with a square lattice. The lattice is defined by the two vectors \mathbf{a}_1 and \mathbf{a}_2 , which are perpendicular and both have magnitude a . The structure is invariant in the z -direction (perpendicular to the plane). For certain values of the spacing, this crystal can have a photonic band gap in the xy plane. Inside this gap, no propagating states are permitted, and the incident light is reflected.

The reciprocal lattice $(\mathbf{b}_1, \mathbf{b}_2)$ is defined such that $\mathbf{a}_i \cdot \mathbf{b}_j = 2\pi\delta_{ij}$. So in this case, $(\mathbf{b}_1, \mathbf{b}_2) = (\frac{2\pi}{a}\mathbf{1}_x, \frac{2\pi}{a}\mathbf{1}_y)$.

As explained in [1], the magnetic field can be described by a function, the Bloch state, that depends on n (the band number), and the wave vector $\mathbf{k} = k_z\mathbf{1}_z + \mathbf{k}_{\parallel}$:

$$\mathbf{H}_{(n, k_z, \mathbf{k}_{\parallel})}(\mathbf{r}) = e^{i\mathbf{k}_{\parallel}\cdot\boldsymbol{\rho}} e^{ik_z z} \mathbf{u}_{(n, k_z, \mathbf{k}_{\parallel})}(\boldsymbol{\rho}) \quad (3.1)$$

Here, $\boldsymbol{\rho} = x\mathbf{1}_x + y\mathbf{1}_y$ and $k_x\mathbf{1}_x + k_y\mathbf{1}_y$. $\mathbf{u}(\boldsymbol{\rho})$ is a periodic function, $\mathbf{u}(\boldsymbol{\rho}) = \mathbf{u}(\boldsymbol{\rho} + \mathbf{R})$,

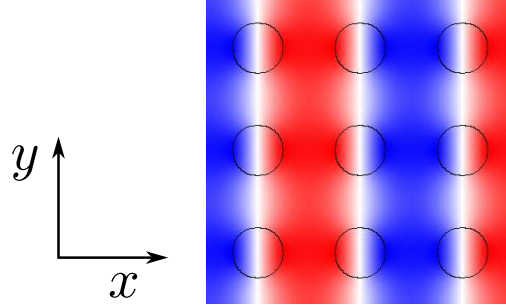


Figure 3.3: Image of the y -component of the magnetic field \mathbf{H} for the first TM mode at the X-point. The structure is a 2D rectangular lattice of rods surrounded by a low-index medium, as shown in Figure 3.2(b).

for $\mathbf{R} = j\mathbf{a}_1 + k\mathbf{a}_2$, where $j \in \mathbb{N}$ and $k \in \mathbb{N}$.

As the Bloch state with wave vector \mathbf{k}_{\parallel} and the Bloch state with wave vector $\mathbf{k}_{\parallel} + j\mathbf{b}_1 + k\mathbf{b}_2$ are identical, we only need to consider a zone for which $-\pi/a < k_x < \pi/a$ and $-\pi/a < k_y < \pi/a$. This is called the Brillouin zone (again, we refer to the book for more information). Given further symmetry properties of the 2D rectangular lattice, we only need to consider three important points in k -space:

1. Γ -point: $\mathbf{k}_{\parallel} = 0$.
2. X-point: $\mathbf{k}_{\parallel} = \mathbf{b}_1 = \pi/a\mathbf{1}_x$
3. M: $\mathbf{k}_{\parallel} = \mathbf{b}_1 + \mathbf{b}_2 = \pi/a\mathbf{1}_x + \pi/a\mathbf{1}_y$

These are also illustrated in Figure 3.11 in the result section, where we calculated the band diagram. From the Γ -point to the X-point we consider waves that propagate in the x -direction, starting from waves with very long wavelength in the x -direction near $\mathbf{k}_{\parallel} = 0$ to waves that have a wavelength corresponding to $\mathbf{k}_{\parallel} = \pi/a$ (the X-point). Since $k = 2\pi/\lambda$ and $k_z = 0$, this corresponds to $\lambda = 2a$, which can clearly be seen from the mode profile in Figure 3.3.

Furthermore, due to the invariance in the z -direction, the modes for a 2D structure can be split in transverse-electric (TE) and transverse-magnetic (TM) modes. For a TE mode, \mathbf{H} is normal to the plane and \mathbf{E} is in the plane. For TM we have just the reverse: \mathbf{E} is normal to the plane and \mathbf{H} is in the plane. In our study we restrict ourselves to waves that propagate in the plane with no k_z -component.

3.1.2 3D vs 2D photonic crystals

In practice, to create a full photonic bandgap in all directions we would need a 3D periodic structure. This is very difficult to fabricate in SOI technology be-

cause of a limit to the number of etching and depositing steps restricts the capability to create arbitrary geometries. A solution for this is to guide light by total internal reflection for the vertical direction (z-axis), and use periodicity in the (x,y) plane. We call this 2D (PhC) + 1D (TIR) structures. This is only efficient if the refractive index of the guiding material is large enough compared to the surroundings, otherwise a lot of light will be scattered upwards and downwards (towards the substrate). The upward scattered light can be used to characterize the photonic crystal cavities in a noninvasive way [2].

An impression of a fabricated 2D photonic crystal is shown in Figure 3.1. To model this system correctly, one should use 3D simulation methods based on the fully vectorial maxwell equations. Luckily, the shape of the cavity resonance is very similar, whether the structure is a 1, 2 or 3 dimensional photonic crystal cavity, so the resulting dynamical effects (in which we are interested in this dissertation) will be roughly the same.

In this dissertation we focus on simulating 2D photonic crystals for several reasons. First, they are mathematically more simple to study than 2D (PhC) + 1D (TIR) devices and we can base our research on extensive studies which have been performed prior to this thesis (such as the work performed by M. Soljacic [3]). In these studies, a rectangular grid of rods was used, with TM polarized light. In principle, it is possible to extend this study using a triangular lattice of holes, with TE polarized light (which is easier to manufacture, especially in SOI technology). In practice this appeared to be nontrivial (finding a good resonance, fitting nonlinearities and so on). Second, doing a 3D simulation with a resonant structure becomes very impractical since this requires a huge amount of computational power. Especially if we want to simulate this system for different input powers and nonlinear materials, simulation times of months become necessary.

3.1.3 Nonlinearities

As explained in section 1.2.2, there are several nonlinear effects that can arise in dielectric structures. These are usually small because they scale with the intensity of the electromagnetic field. However, in a cavity there is a huge amplification of the field intensity, and the nonlinear effects can become relevant. In the rest of this chapter, we describe what happens when we include these nonlinearities. For this dissertation, we have focused on the Kerr nonlinearity, which is easy to model both in descriptive models and in an FDTD simulation. The reason the Kerr nonlinearity is easy to model in FDTD is because we can assume it is instantaneous, so we can simply include it in the refractive index of the material. The behavior of slower nonlinearities (e.g., thermal nonlinearities) is often qualitatively very similar.

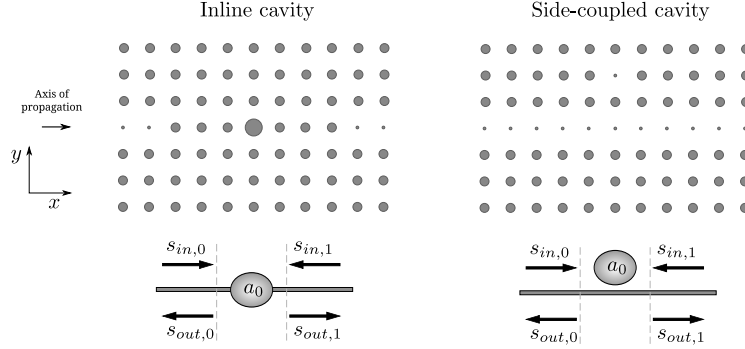


Figure 3.4: Illustration of the two type of cavities that are used throughout this dissertation. Top: a physical 2D layout of the cavity. Bottom: the schematic representation. Left: an inline coupled cavity. Right: a side-coupled cavity.

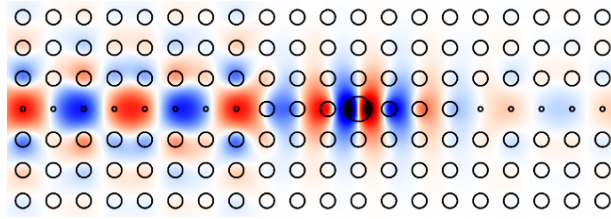


Figure 3.5: Exciting the cavity mode using a point source in an FDTD simulation.

3.1.4 Photonic crystal waveguides and cavities

By using linear defects, we can guide light from one location to another. The basic idea is to create a waveguide by modifying a linear sequence of unit cells, as shown schematically in Figure 3.2(a) and Figure 3.4. Light that propagates in the waveguide with a frequency within the band gap of the crystal is confined to the defect, and can be directed along it.

A cavity can be created by introducing a point defect, as shown schematically in Figure 3.4 and illustrated with an FDTD simulation in Figure 3.5. Theoretically we need an infinite number of layers of the bulk crystal around the cavity to stop light from propagating through these layers. By using only a few layers, light can tunnel through these layers and enter the cavity region. Light is still confined inside the cavity, as it is surrounded by the rectangular lattice. For this 2D topology, the cavity supports a dipole-type localized resonant mode [3].

An optical resonator can be inside the guiding structure, or next to it. If the cavity is inside the guiding structure, it represents an inline coupled cavity,

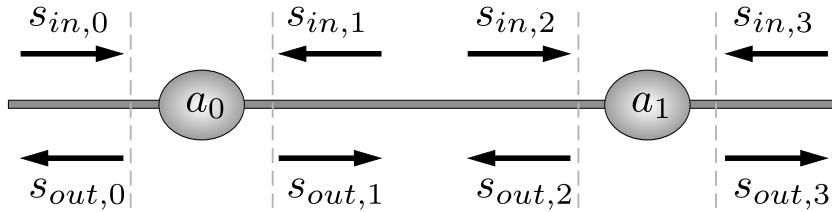


Figure 3.6: A schematic representation of two coupled cavities in series. The reference planes are chosen symmetrical on both sides of the cavity. The distance from the center of the cavity to the reference plane determines the phase reflection parameter ϕ_k .

which we will use mostly in this dissertation. If the cavity is outside the guiding structure, it represents a side-coupled cavity [4]. In Figure 3.4, we show an example 2D structure and the schematic representation for both cases.

3.2 Simulation of photonic crystal cavities

In this section we first explain the Coupled Mode Theory (CMT), a descriptive model for the photonic crystal cavity. This allows us to gain insight in the steady-state characteristics and nonlinear dynamics of the cavities. After this, we compare this approximate model with a full simulation of a 2D photonic crystal (as explained in the beginning of this chapter) using the FDTD method (see 4.1). We design a photonic crystal waveguide and a proper cavity, match the nonlinear behavior of a single cavity to the CMT model, and then look at a series of two coupled resonators. The non-trivial dynamical system of two coupled resonators in FDTD is captured very accurately with the Coupled Mode Theory, which means the Coupled Mode Theory can be used to simulate large photonic reservoirs without sacrificing much accuracy. In the later chapters, we therefore use the CMT equations to simulate photonic reservoirs.

3.2.1 Coupled Mode Theory

The Coupled Mode Theory offers an elegant way to dynamically describe passive optical resonators [5, 6]. This theory shows good correspondence with experiments of such passive resonators [7–9], and it often results in models which are still analytically solvable, both for varying power and varying wavelength of the input light.

3.2.1.1 Equations

Figure 3.6 shows the schematic representation of two coupled inline cavities in series.

To model this geometry we use the Coupled Mode Theory as developed by H. Haus and co-workers [10]. The time-domain evolution of the cavity energies (a_k) is given by:

$$\frac{da_k}{dt} = \left[j(\omega_{r,k} - \omega + \delta\omega_k) - \frac{1}{\tau} \right] a_k + d_k s_{in,2k} + d_k s_{in,2k+1}, \quad (3.2)$$

and the output is given by

$$s_{out,2k} = \exp(j\phi_k) s_{in,2k} + d_k a_k, \quad (3.3)$$

$$s_{out,2k+1} = \exp(j\phi_k) s_{in,2k+1} + d_k a_k, \quad (3.4)$$

for inline coupled cavities, and

$$s_{out,2k} = \exp(j\phi_k) s_{in,2k+1} + d_k a_k, \quad (3.5)$$

$$s_{out,2k+1} = \exp(j\phi_k) s_{in,2k} + d_k a_k, \quad (3.6)$$

for side-coupled cavities. $k = 1, \dots, N$; with N the number of cavities (N is not limited to 2 as shown in the figure). The frequency of the input light is given by ω , while the resonance frequency of each cavity is given by $\omega_{r,k}$. Furthermore $d_k = j \exp(j\phi_k/2)/\sqrt{\tau}$, where τ is the lifetime of the cavity and ϕ_k represents a phase shift due to the finite distance d between the reference plane and the cavity, and also depends on the resonator mirror reflection properties. Although the qualitative behavior for one cavity does not depend on this phase shift, the collective behavior of two or more coupled cavities can fundamentally change by modifying ϕ_k . In the discussion that follows, we assume there is no additional phase shift nor loss between the reference planes of the two cavities (i.e. $s_{in,2} = s_{out,1}$). The nonlinear frequency shift is $\delta\omega_k = -|a_k|^2/(P_0\tau^2)$, with P_0 the ‘characteristic nonlinear power’ of the cavity [3]. In these equations $|a_k|^2$ is the energy in the cavity mode. $|s_{in,2k}|^2$ (resp. $|s_{in,2k+1}|^2$) represents the power flowing in the (single-mode) waveguide in the forward (resp. backward) direction. Thus, $|s_{in,0}|^2 \equiv P_{in}$ is the input power, $|s_{out,2N+1}|^2 \equiv P_{trans}$ is the transmitted power.

We can furthermore define the detuning of the cavity as:

$$\Delta_k = (\omega - \omega_{r,k}) \tau_k \quad (3.7)$$

This dimensionless parameter describes how far we are removed from resonance.

3.2.1.2 A single cavity

We first study a single inline cavity, the steady-state behavior and the phenomenon of bistability. Since we only have one cavity, we omit the cavity number suffix and define $s_{in} = s_{in,0}$ and $s_{out} = s_{out,1}$. $P_{in} = |s_{in}|^2$ is the input power entering the cavity from the left, and $P_{out} = |s_{out}|^2$ is the output power on the right of the cavity. $|a|^2$ is the optical energy inside the cavity and has units of J .

When the system is in steady-state, the left hand side of equation 3.2 becomes zero, and we can write:

$$0 = \left[j(\omega_r - \omega + \delta\omega) - \frac{1}{\tau} \right] a_0 + d \cdot s_{in} \quad (3.8)$$

Furthermore we know that $\Delta = (\omega_r - \omega) \tau$, and $\delta\omega = -|a|^2 / (P_0 \tau^2)$, so

$$0 = \left[j \left(\frac{\Delta}{\tau} - \frac{|a|^2}{P_0 \tau^2} \right) - \frac{1}{\tau} \right] a_0 + d \cdot s_{in} \quad (3.9)$$

$$d \cdot s_{in} = \frac{1}{\tau} \left[j \left(-\Delta + \frac{|a|^2}{P_0 \tau} \right) + 1 \right] a \quad (3.10)$$

Also, $s_{out} = d \cdot a$ and $d = j e^{j\phi/2} / \sqrt{\tau}$, so

$$d \cdot s_{in} = \frac{1}{\tau} \left[j \left(-\Delta + \frac{|s_{out}|^2}{P_0 \tau |d|^2} \right) + 1 \right] s_{out} / d \quad (3.11)$$

$$-\frac{e^{j\phi}}{\tau} \cdot s_{in} = \frac{1}{\tau} \left[j \left(-\Delta + \frac{|s_{out}|^2}{P_0} \right) + 1 \right] s_{out} \quad (3.12)$$

If we then multiply both sides with their complex conjugate, we get

$$P_{in} = \left[1 + \left(\Delta - \frac{P_{out}}{P_0} \right)^2 \right] P_{out} \quad (3.13)$$

$$\frac{P_{out}}{P_{in}} = \frac{1}{1 + (\Delta - P_{out}/P_0)^2} \quad (3.14)$$

This steady-state is shown for different detunings in Figure 3.7. If the detuning $\Delta = (\omega_r - \omega) \tau$ becomes larger than $\sqrt{3}$, there is bistable behavior, which means there are several stable steady-state solutions for a given input power.

In the linear case, $\delta\omega = 0$. If we further substitute $\Delta = (\omega_r - \omega) \tau$, we get the standard Lorentzian curve which we will use later on to fit with the FDTD results:

$$\frac{P_{out}}{P_{in}} = \frac{1/\tau^2}{1/\tau^2 + (\omega - \omega_r)^2} \quad (3.15)$$

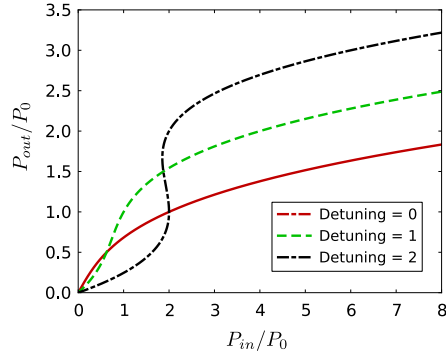


Figure 3.7: Steady-state curves of a single cavity with Kerr nonlinearity defined by a characteristic power P_0 .

3.2.1.3 Dynamics of two coupled cavities in serie

In this section we focus on the dynamical behavior of two identical cavities that are coupled in series as shown in Figure 3.6. Whenever the parameters for the two cavities are identical, we again omit the index, i.e. $\phi_k = \phi$, $d_k = d$ and so on. For a dynamical system, a linear stability analysis reveals whether or not the system is stable. This is done by examining the eigenvalues of the system's Jacobian. If all eigenvalues have a negative real part, the system is stable. In some cases, an unstable fixed point implies chaos or self-pulsation. To distinguish between both, one can calculate the system's maximal Lyapunov exponent. If this exponent is larger than zero, the system is chaotic. For a stable periodic solution, the maximal Lyapunov exponent is zero. This is elaborated in detail in our paper [5], and the stability analysis for this system of two cavities is summarized in Figure 3.8.

For example: from Figure 3.8(a) we see that a series of two resonators will self-pulsate when $\phi = 0.2\pi$, $P_{in} = P_0$ and $\Delta = 2$. The resulting P_{out} for this system is shown in Figure 3.9.

For larger circuits with arbitrary topology, it becomes very cumbersome to evaluate the Jacobian and calculate the largest Lyapunov exponent. Just as an illustration, we show the dynamics of a series of three coupled cavities in 3.10. We can see that, depending on the input power, we can have different sorts of dynamics, ranging from stable to self-pulsing to chaos.

In the rest of this dissertation, we study even larger networks. Therefore, we have created a mathematical framework and the accompanying software, called *Caphe*, to study arbitrary topologies consisting of components with dynamical nonlinear behavior. This is a prerequisite to simulating a nanophotonic reservoir and is elaborated on in chapter 4.

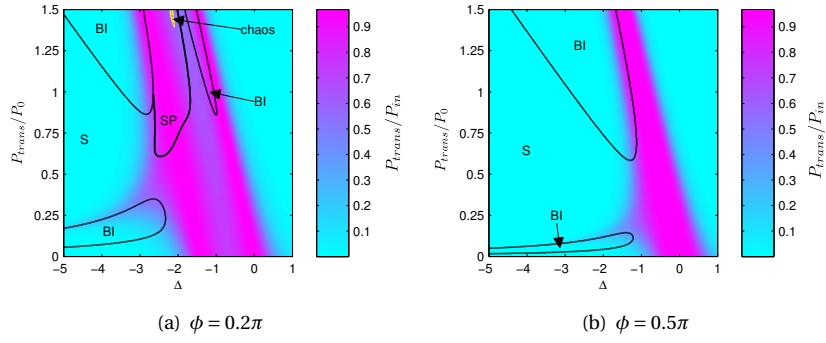


Figure 3.8: Transmission (P_{trans}/P_{in}) and classification of the dual-cavity device for different ϕ . Different regimes are indicated: (S) stable, (BI) bistable, (SP) self pulsing and chaos. Note: in this figure the definition of detuning is chosen opposite, i.e., $\Delta = (\omega - \omega_r) \tau$.

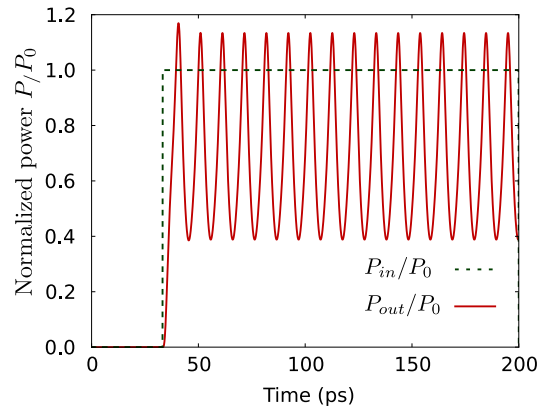


Figure 3.9: Example time-trace for $\phi = 0.2\pi$, $P_{in} = P_0$ and $\Delta = 2$.

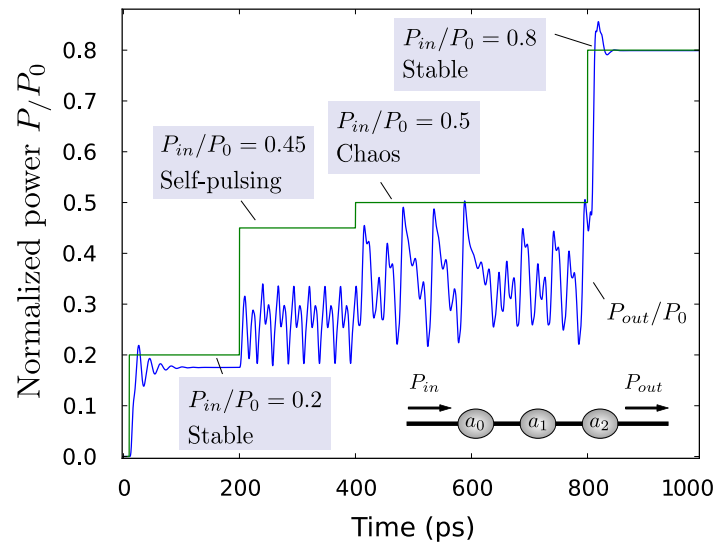


Figure 3.10: Time-trace for a series of 3 cavities (shown in inset). $\phi = 0.5\pi$ and $\Delta = 0.75$. Depending on the input power, the dynamics can range from stable to self-pulsing to chaos. This demonstrates the rich dynamics of a nonlinear dynamical system consisting of photonic crystal cavities. In the following chapters, we will make networks of 100s of resonators, and predicting the regions of stability becomes very difficult.

3.2.2 FDTD

In this section we show simulation results for a square lattice of dielectric columns (see Figure 3.2(a)). We use the values supplied by the paper of M. Soljacic [3] as a starting point: the rods have a radius of $r = 0.25a$, and the refractive index of the rods is 3.5 while the surrounding medium has a refractive index of 1.5. These values are used to compare the FDTD method with the Coupled Mode Theory, and we show that it is possible to describe this very complex system with only a few variables and parameters.

3.2.2.1 2D photonic crystal with a rectangular lattice

Figure 3.11 shows the calculated bandgap of the 2D rectangular lattice (see inset (1)). The lattice constant a allows us to use dimensionless variables. The normalized frequency is defined by:

$$\omega_n = \frac{\omega a}{2\pi c} \quad (3.16)$$

While the k -vector is given by $2\pi/a$. The horizontal axis shows the value of the in-plane wave vector k_{\parallel} . As we move from left to right, k moves along the triangular edge of the irreducible Brillouin zone, from Γ to X to M, as shown in inset (2) of Figure 3.11.

Fully-vectorial eigenmodes of Maxwell's equations with periodic boundary conditions were computed by preconditioned conjugate-gradient minimization of the block Rayleigh quotient in a planewave basis, using a freely available software package called MPB [11].

The modes for TE and TM can be completely different (recall that the TM polarized modes have an electric field that is perpendicular to the plane, and a magnetic field along the plane). Between the different bands of the TM-polarized modes, there is a region where no light can propagate, which means it has a complete bandgap for the TM polarization (this is explained in more detail in the book on photonic crystals [1]). For the simulated structure, there is a bandgap from $\omega_n=0.2424$ to $\omega_n=0.2897$. We can use this to trap light inside a so-called photonic crystal cavity.

3.2.2.2 2D photonic crystal waveguide and cavity

To find the resonance wavelength of a cavity we have used the freely available FDTD simulator called MEEP [12]. The parameters for the FDTD simulation which we performed are listed in table 3.1.

Since it is very difficult to calculate the mode profile with the FDTD simulator, we will use a simple point source to excite the waveguide. This has the drawback, however, that it creates a lot of unwanted scattering. The solution

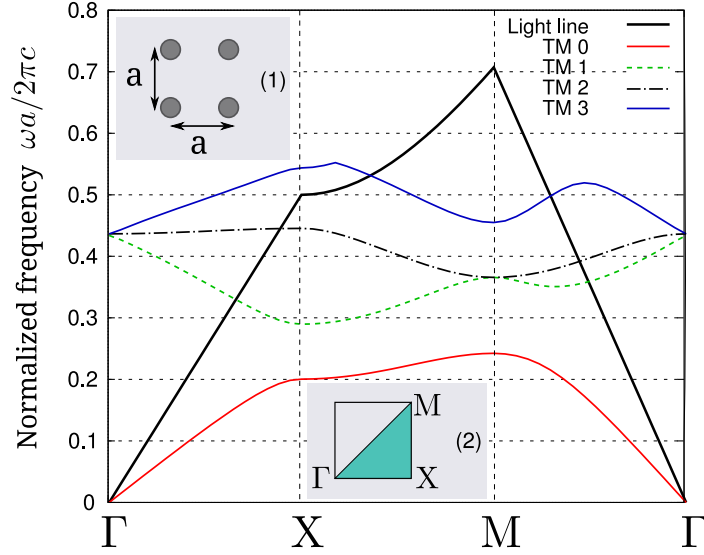


Figure 3.11: Banddiagram for a 2D photonic crystal with a rectangular lattice (shown in inset (1)) for TM polarization. We traverse the k-space in the 2D plane. One can see that there is a region of frequencies where no light can propagate for any direction (the bandgap). Also, above the light line, no light can propagate. The simulations were performed with MPB [11]. Inset (2) shows the irreducible Brillouin zone for this structure.

Name	Description	Value
a	Lattice constant (unit cell in the FDTD simulation)	1
r	Radius of the rods	0.25a
n_{rod}	Refractive index of the rods	3.5
n_{medium}	Refractive index of the surrounding medium	1.5
ω_n	Source center frequency (normalized)	0.2615
$d\omega_n$	Source normalized resonance width	0.008
n_2	Kerr coefficient	$1.5 \cdot 10^{-5} \mu m^2/W$

Table 3.1: Values used for the time-domain FDTD simulations.

to this problem is to perform two simulations: a reference simulation without cavity and a simulation with cavity. Dividing the resulting spectra results in the correct transmission. For now, we do not use nonlinear effects so we put $n_2 = 0 \mu m^2/W$. Below we explain the procedure in slightly more detail:

1. Reference simulation: this simulation is performed without the cavity. This means we exactly simulate the structure of Figure 3.2(a). The input is a point source with a gaussian-shaped frequency profile that is launched at the left of the simulation window. We measure the output in a flux plane at the right of the simulation window: $F_{ref}(\omega_n)$.
2. Cavity simulation: we use the same source, the same flux plane, but we add a cavity in the center of the simulation window. This simulation takes much longer as the resonance causes light to slowly die out. The output is $F_{cav}(\omega_n)$. Figure 3.5 shows the simulation where one can clearly see that the mode is excited.
3. Calculate the normalized transmission through the cavity:
 $T(\omega_n) = F_{cav}(\omega_n)/F_{ref}(\omega_n)$. This function is then fitted to the Lorentzian shape, which we derived in section 3.2.1.2. We repeat the equation here in a slightly modified form which we actually used to perform the fitting):

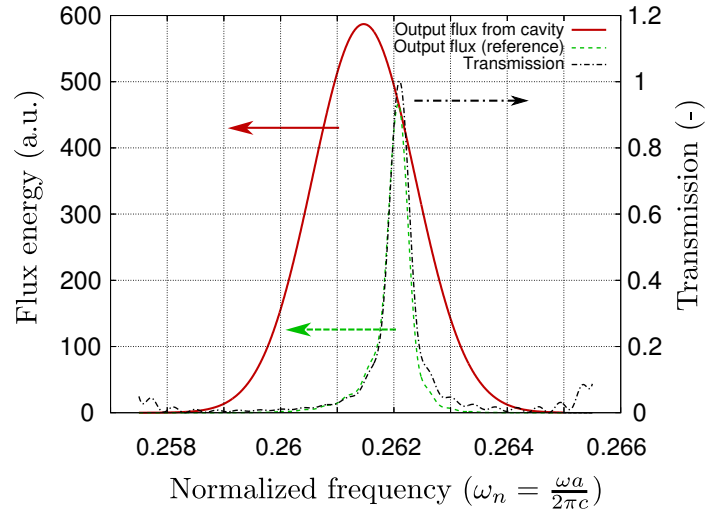
$$L(\omega_n) = \frac{A\gamma_n^2}{\gamma_n^2 + (\omega_n - \omega_{r,n})^2} \quad (3.17)$$

Here, ω_r is the resonance frequency of the system, and $\gamma_n = 1/\tau_n$ is the (normalized) linewidth of the resonance. The linewidth is inversely proportional to the Q-factor of the system and is given by: $Q = \omega_{r,n}/2\gamma_n = \omega_r/2\gamma$. Figure 3.12 shows the resulting spectrum. For our system we find after fitting that $\omega_{r,n}=0.262087$, and $\gamma=0.000204194$. This results in a Q-factor of 641.76.

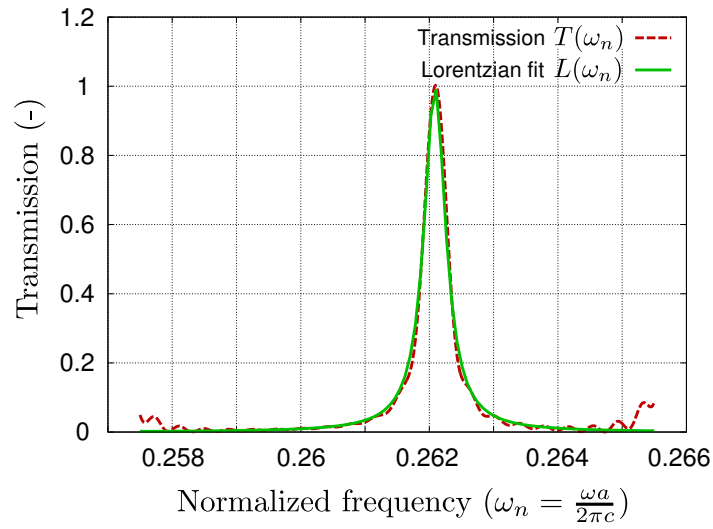
3.2.2.3 Behavior of a photonic crystal cavity with Kerr nonlinearity

Because of the resonance, high powers can be stored inside a cavity. This also means that nonlinear effects (that typically depend on some higher order power of the electric field) are greatly enhanced inside the resonator. In this dissertation, we use the Kerr effect (parameter n_2) as dominant nonlinear effect (see 1.2.2). For the following simulations, we used $n_2 = 1.5 \cdot 10^{-5} \mu m^2/W$. By adding this nonlinear term to the rods, we were able to observe a bistable behavior, similar to Figure 3.7.

It is very important to know how good the 2D photonic crystal cavity actually matches the CMT behavior. If the behavior is very similar, we do not need



(a) Output fluxes of the reference and cavity simulation.



(b) Normalized transmission through the cavity, and the resulting fit to the Lorentzian curve.

Figure 3.12: Finding the resonance in the photonic crystal cavity using a 2D FDTD simulation. The resonance is of a Lorentzian shape, and by fitting this theoretical curve (equation 3.17) to the normalized transmission we can calculate the linewidth and resonance wavelength of the photonic crystal cavity.

the very intensive FDTD simulations anymore to study large networks of resonators.

To check the similarity, we calculate, using FDTD, the transmission for different input powers. This means we launch an input pulse with a certain power into the photonic crystal waveguide and then observe how much power is measured behind the cavity. We repeat this simulation for different input powers to obtain a part of the bistability curve (represented by the blue crosses in Figure 3.13). To get to the upper branch of the curve, we need an additional 'trigger' pulse (see the circle on Figure 3.13, in our simulation this is a very short pulse superimposed on the CW signal), and measure the output power after the trigger pulse has died out. We then fit the analytical formula from equation 3.15 to the result and obtain a detuning of $\Delta = -3.4411$. The good correspondence between FDTD and the analytical model confirms that the Lorentzian-shaped resonance is indeed a good approximation of the physical reality. The specific value of P_0 depends on n_2 . For a realistic value n_2 of $1.5 \cdot 10^{-5} \mu\text{m}^2/W$, P_0 can be brought below 77 mW [3].

In practice however, it is very difficult to observe this behavior. This is mainly due to losses inside the cavity which trigger other effects such as free carrier absorption and heating of the material. Both have the effect of changing the refractive index of the material, as explained in 1.2.2. The time scale for the temperature effect ($\sim \mu\text{s}$) is several orders of magnitude slower than the Kerr effect ($\sim \text{fs}$), which means that if we can measure fast enough and $P_0 < P_{laser}$, we can in principle observe the Kerr effect. In practice, linear losses will have a detrimental effect on the nonlinear effects, as shown by T. Van Vaerenbergh in [6]. Also, the plasma effect (caused by free carriers), induces a blueshift of the resonance and counteracts the Kerr effect. When using a longer wavelength, the two photon absorption, and hence the generation of free carriers, is reduced. Also, free carriers can be extracted from the ring using a PIN junction. In this way, we might be able to experimentally isolate the Kerr effect. Experimental measurements of these nonlinearities have been performed in a similar photonic crystal cavity structure [13], or in Si_3N_4 ring resonators [14].

3.2.2.4 2 cavities in a row

The final step in comparing the CMT to FDTD is simulating a series of 2 coupled cavities. This structure is represented schematically in Figure 3.6 and a 2D FDTD simulation is shown in Figure 3.14. We use the same parameters for the geometry of the system, but now the first cavity is coupled to a second cavity. The purpose is to reproduce the self-pulsing which we observed in the simplified model, and of which we showed the conditions in section 3.2.1.3.

We get an extra degree of freedom here: the number of rods between the two cavities. This determines the phase ϕ , and greatly influences the dynamics

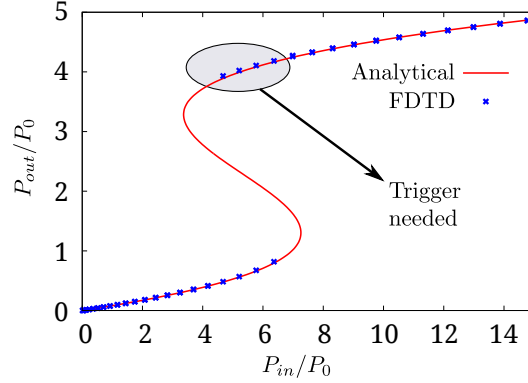


Figure 3.13: Bistable curve for a nonlinear photonic crystal cavity. The blue dots are the result of the FDTD simulations. This is fitted to the analytical formula for a lorentzian-shaped resonance with nonlinearity (full line in red).

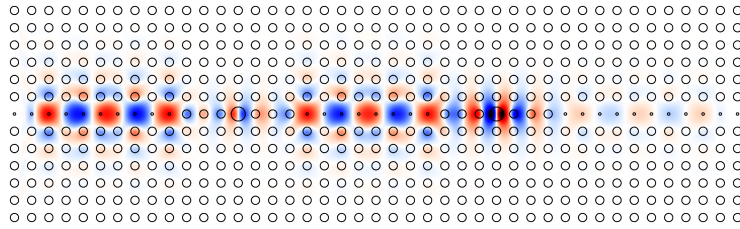


Figure 3.14: Ez-field of the FDTD simulation of two cavities in a row after 170 optical periods. The system is self-pulsing. Simulation parameters: $P_{in} = 5.6254/P_0$, $d = 14$ (number of rods between the two cavities) and $\Delta = 2.225$.

in the two cavities. We choose $d = 14$, or 14 rods between the centers of the two cavities. The phase difference ϕ determines the distance between the two peaks in the linear transmission of two cavities [5]. This linear transmission can be calculated for a given Δ using FDTD simulations, and this transmission should be the same as predicted by CMT for a certain ϕ . In this way we fit ϕ to be $\phi = 0.2314$.

For this set of parameters, coupled mode theory predicts the component to start self-pulsing at around $P_{in} = 5.52/P_0$. The exact power at which self-pulsing starts is quite ambiguous since there is a region just before self-pulsing where the output power is also pulsing, but is damping out to a steady-state value. From our simulations in Figure 3.14, we will use $P_{in} = 5.6254/P_0$. Figure 3.14 shows a field plot after 170 optical periods.

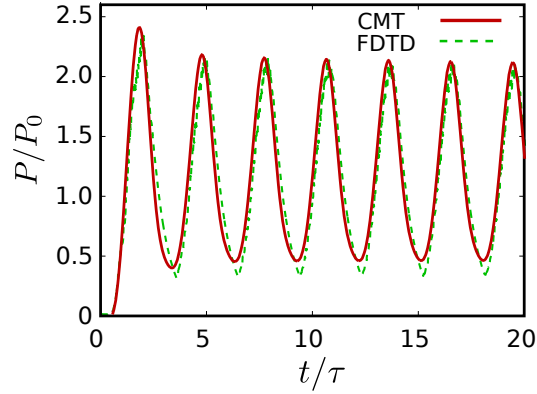


Figure 3.15: Comparison of the CMT and FDTD simulation. The simplified CMT model is able to accurately describe the physical behavior of two coupled nonlinear resonators.

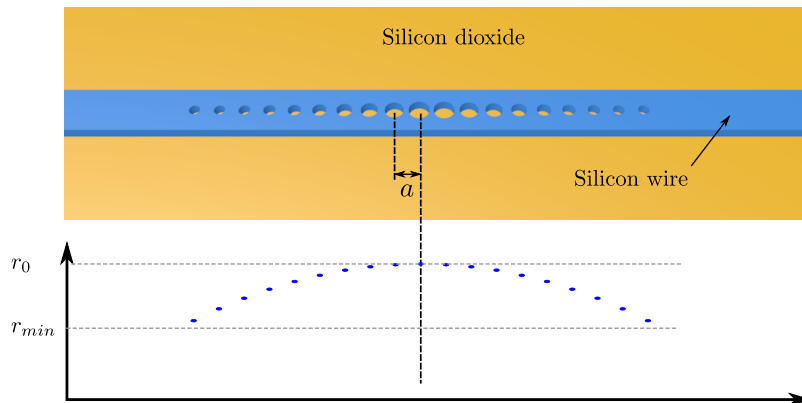


Figure 3.16: A 1D wire cavity. The radius increases towards the center according to a parabolic profile.

We find that apparently there is still a small error in predicting ϕ , since the component is not yet self-pulsing in CMT for the used P_{in} , whereas it does self-pulsate in FDTD. When increasing ϕ slightly to 0.2333, we can reduce the threshold for self-pulsing. With these adjusted values, we get a reasonable match between the CMT and the FDTD time traces, as shown in Figure 3.15.

3.3 Measurements

Although in the previous section we described a 2D rectangular topology, there are other ways to create a resonant cavity using the principles of photonic crystals. One-dimensional Photonic Crystal Cavities (PhCC) (like the wire cavity in Figure 3.16) are attractive because of their structural simplicity and smaller size compared with higher dimensional PhCCs. There are many applications for these structures such as a thermo-optic optical switch [15], opto-mechanical devices [16], nano-sensors [17] and so on. Recently, it has been demonstrated independently by E. Kuramochi [18] and Q. Quan et al [19] that one can create very high Q-factor photonic crystal cavities by creating holes in an SOI wire waveguide.

There are several ways to fabricate photonic crystal cavities. The by far most used technique is e-beam, since it is one of the few techniques that fulfills the accuracy requirements on these length scales. However, due to its direct sequential writing, e-beam lithography is time consuming for large area structures. So although photonic crystals can be made with high accuracy using e-beam lithography [20], [21], this does not allow us to efficiently create a large network of interconnected cavities. Especially since the area in which e-beam is performed is relatively small, one will have to reposition the structure several times during fabrication, and stitching errors will arise that cause unwanted reflections.

There are other methods that are faster, such as Laser Interference Lithography [22], but these have complex fabrication procedures, where two interfering laser beams create a pattern onto the photo resist, after which the substrate has to be rotated, and the thickness of the resist has to be controlled very accurately.

We base our fabrication on the standard SOI technology and use lithography techniques as described in 1.2. As this technology becomes increasingly accurate, it becomes possible to create relatively high Q-factor cavities, as we show further on. The complete fabrication process of our 1D cavity prototypes has been done on a 300mm SOI wafer using a CMOS pilot line at imec with 193 nm deep ultraviolet immersion lithography. Note that this is different from the standard 200 mm process from ePIXfab [23], which is used more often. The results shown here are a first characterization of photonic crystal cavity devices which were fabricated on a 300mm SOI wafer.

3.3.1 Design and fabrication

To design a 1D wire cavity, we have used the structure as proposed in [24], which is illustrated in Figure 3.16. The radius of the holes $r(i)$ is given as a parabolic function:

$$r(i) = \max[r_{min}, r_0(1 - (i/m)^2)] \quad (3.18)$$

Parameter	Description	Value
a	Lattice constant	$0.4\mu m$
r_0	Radius of the largest hole	$0.3a = 0.12\mu m$
r_{min}	Minimum hole size	$0.22a = 0.088\mu m$
m	Radius parameter	21
i	Hole index	[-14..14]

Table 3.3: Parameters used for the fabrication of the 1D wire cavity.

Because the effective index is slightly different from the designs in [24], and because we know that the fabricated hole size depends to a large extent on the dose strength during processing, we have put several variations of this design on the mask. These variations are:

1. Using a modified radius $r_m(i) = F \cdot r(i)$, for $F \in [0.8, 0.9, 1, 1.1]$.
2. Using different waveguide widths $w \in [0.46, 0.48, 0.50, 0.52, 0.54, 0.56, 0.58]\mu m$.

3.3.2 Measurement and post-processing

The fabricated components have been measured using a semi-automatic measurement setup, which allows us to automatically move over the different structures without manual re-alignment. Thanks to this system, it was possible to measure these many variations in a relatively short period of time.

There are several important parameters which we want to extract from the raw spectral data:

1. The resonance wavelengths λ_r . These can be found by performing a simple peak detection. There can be multiple peaks in the spectrum.
2. The Q-factor, or quality factor, of the resonator. The higher the Q, the longer light stays inside the resonator. This can be calculated by fitting the spectrum to the Lorentzian curve, as we have done before. Here, we opt to fit to the logarithmic spectrum, because the result will be more accurate.

$$L_{log}(\lambda) = 10 \cdot \log_{10} \left(\frac{A(\lambda_{3dB}/2)^2}{(\lambda - \lambda_{res})^2 + (\lambda_{3dB}/2)^2} \right) \quad (3.19)$$

λ_{3dB} is the full width at half maximum (FWHM) of the resonance. The Q-factor is then equal to $Q = \lambda_r / \lambda_{3dB}$.

3. The finesse of the resonator. This is defined as the free spectral range divided by the full width at half maximum (FWHM) bandwidth of the resonator:

$$\mathcal{F} = \frac{Q \cdot FSR}{\lambda_{3dB}} \quad (3.20)$$

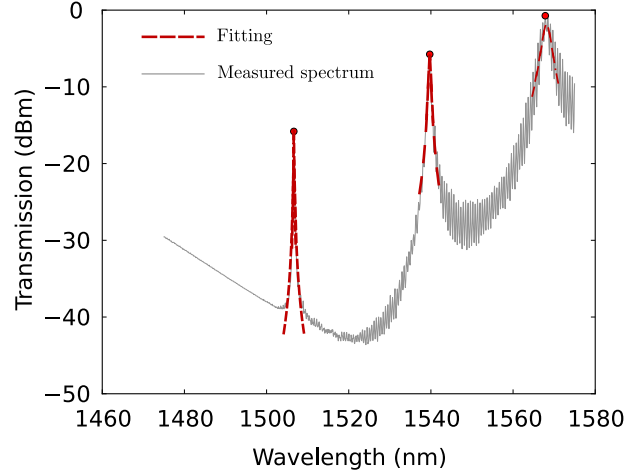


Figure 3.17: Measurement of the photonic crystal cavities with $w_{wg} = 0.46\mu\text{m}$, $F = 1.1$. Three distinct resonances are found, with a maximum Q-factor of 5067 (the left resonance). See Table 3.4 for more information.

The finesse is only determined by the loss in the resonator. The higher the finesse, the sharper the peaks while still maintaining a good (large) FSR. High finesses are useful for spectrometers, that benefit from a small bandwidth (high spectral resolution) and large FSR (large spectral range for measuring).

3.3.3 Results

Figure 3.17 shows the result from detecting peaks and fitting them to a Lorentzian curve to extract the Q-factors. The final measurement results are shown in Table 3.4. The insertion loss is defined as the actual loss from the device itself. To remove the influence of fiber-chip, chip-fiber and waveguide losses, we measured the same structure without a cavity (i.e., a reference waveguide), and normalized the spectrum to this measurement.

For some measurements with high-Q cavities, we see a red shift of the spectrum. This is a nonlinear effect most likely caused by temperature.

3.3.4 Acknowledgements

The author wishes to acknowledge imec to provide the opportunity to design test-chips for the 300 mm wafer run, which was the first step towards all measurements which were reported in this section. Furthermore, the author

F	λ_r	Q-factor	finesse	Insertion Loss (dB)
0.8	1564.44	5097		3.4
0.9	1533.88	9238	149.2	14
	1559.67	2914	49.1	1.4
1	1490.99	18629	376.6	20.6
	1520.92	6526	129.4	6.6
	1548.75	1336	26	2.3
1.1	1479.92	5067	112.8	18.2
	1512.91	2092	45.6	6.8
	1542.02	566	12.1	3

Table 3.4: Measurement results for the 1D wire cavity for $w_{wg} = 460\mu m$.

wishes to acknowledge Shankar Kumar Selvaraja to perform the processing, and Weiqiang Xie for doing most of the measurements.

3.4 Conclusions

In this chapter we have made a detailed analysis of the photonic crystal cavity which we use as a basic building block for nanophotonic reservoir computing. These cavities can be simulated with approximate models using only a single dynamical variable, called the coupled mode theory (CMT). We have shown in this chapter that these approximate models are still valid when coupling two of these resonators, by comparing them with accurate finite difference time domain (FDTD) simulations. In particular, we were able to observe self-pulsation in a series of two cavities, in both the simplified CMT model and in an FDTD simulation. By tuning the parameters, we were able to match the time-trace of both simulation methods. In later chapters we will further investigate how we can use a large network of photonic crystal cavities to solve a speech recognition task (chapter 5) and a signal generation task (chapter 6).

Photonic crystal cavities are also useful outside the field of reservoir computing. When coupling three of these resonators, chaos is quite common at low powers. This could be used for integrated random bit generators [25] or chaos communication [26].

References

- [1] J. D. Joannopoulos, S. G. Johnson, J. N. Winn, and R. D. Meade. *Photonic Crystals: Molding the Flow of Light*. Princeton University Press, 2008.

-
- [2] A. Avoine, C. Vion, J. Laverdant, S. Bonnefont, O. Gauthier-Lafaye, L. Coolen, and a. Maître. *Photonic crystal cavity modes in the visible range characterized by scattering spectroscopy*. Physical Review A, 82(6):1–7, December 2010.
- [3] M. Soljagic, M. Ibanescu, S. G. Johnson, Y. Fink, and J. D. Joannopoulos. *Optimal Bistable Switching in Nonlinear Photonic Crystals*. Phys. Rev. E, 66:055601, 2002.
- [4] Andrey Miroschnichenko, Yuri Kivshar, Christoph Etrich, Thomas Pertsch, Rumén Iliev, and Falk Lederer. *Dynamics and instability of nonlinear Fano resonances in photonic crystals*. Physical Review A, 79(1):1–7, 2009.
- [5] Bjorn Maes, Martin Fiers, and Peter Bienstman. *Self-pulsing and chaos in series of coupled nonlinear micro-cavities*. Physical Review B11, 7911(111), 200911.
- [6] T. Van Vaerenbergh, M. Fiers, P. Mechet, T. Spuesens, R. Kumar, G. Morthier, B. Schrauwen, J. Dambre, and P. Bienstman. *Cascadable Excitability in microrings*. Optics Express, 20(18):20292–20308, 2012.
- [7] Thomas J Johnson, Matthew Borselli, and Oskar Painter. *Self-induced optical modulation of the transmission through a high-Q silicon microdisk resonator*. Optics express, 14(2):817–31, January 2006.
- [8] Q. Lin, T. J. Johnson, C. P. Michael, and O. Painter. *Adiabatic self-tuning in a silicon microdisk optical resonator*. Opt. Express, 16(19):14801–14811, Sep 2008.
- [9] Wolfram H. P. Pernice, Mo Li, and Hong X. Tang. *Time-domain measurement of optical transport in silicon micro-ring resonators*. Opt. Express, 18(17):18438–18452, Aug 2010.
- [10] C Manolatou, MJ Khan, SH Fan, PR Villeneuve, HA Haus, and JD Joannopoulos. *Coupling of modes analysis of resonant channel add-drop filters*. IEEE Journal of Quantum Electronics, 35(9):1322–1331, 1999.
- [11] Steven G. Johnson and J. D. Joannopoulos. *Block-iterative frequency-domain methods for Maxwell's equations in a planewave basis*. Opt. Express, 8(3):173–190, 2001.
- [12] Ardavan F. Oskooi, David Roundy, Mihai Ibanescu, Peter Bermel, J. D. Joannopoulos, and Steven G. Johnson. *MEEP: A flexible free-software package for electromagnetic simulations by the FDTD method*. Computer Physics Communications, 181:687–702, January 2010.

- [13] P. Barclay, K. Srinivasan, and O. Painter. *Nonlinear response of silicon photonic crystal microresonators excited via an integrated waveguide and fiber taper*. Optics Express, 13(3):801–820, 2005.
- [14] Kazhurio Ikeda, Robert E. Saperstein, Nikola Alic, and Yeshaiah Fainman. *Thermal and Kerr nonlinear properties of plasma-deposited silicon nitride / silicon dioxide waveguides*. Optics Express, 16(17):12987–12994, 2008.
- [15] Laurent-Daniel Haret, Takasumi Tanabe, Eiichi Kuramochi, and Masaya Notomi. *Extremely low power optical bistability in silicon demonstrated using 1D photonic crystal nanocavity*. Optics Express, 17(23):21108–21117, Nov 2009.
- [16] Matt Eichenfield, Ryan Camacho, Jasper Chan, Kerry J. Vahala, and Oskar Painter. *A picogram- and nanometre-scale photonic-crystal optomechanical cavity*. Nature, 459(7246):550–555, 2009.
- [17] Tsan-Wen Lu, Yi-Hua Hsiao, Wei-De Ho, and Po-Tsung Lee. *Photonic crystal heteroslab-edge microcavity with high quality factor surface mode for index sensing*. Applied Physics Letters, 94:141110, 2009.
- [18] Eiichi Kuramochi, Takasumi Tanabe, Hideaki Taniyama, Kohei Kawasaki, and Masaya Notomi. *Ultrahigh-Q silicon-on-insulator one dimensional mode-gap nanocavity*. In Lasers and Electro-Optics (CLEO) and Quantum Electronics and Laser Science Conference (QELS), 2010 Conference on, pages 1–2, may 2010.
- [19] Qimin Quan, Parag B. Deotare, and Marko Loncar. *Photonic crystal nanobeam cavity strongly coupled to the feeding waveguide*. Applied Physics Letters, 96:203102, 2010.
- [20] Eiichi Kuramochi, Masaya Notomi, Satoshi Mitsugi, Akihiko Shinya, Takasumi Tanabe, and Toshifumi Watanabe. *Ultrahigh-Q photonic crystal nanocavities realized by the local width modulation of a line defect*. Applied Physics Letters, 88(4):041112, 2006.
- [21] M Devittorio. *Two-dimensional photonic crystal waveguide obtained by e-beam direct writing of SU8-2000 photoresist*. Microelectronic Engineering, 73-74:388–391, 2004.
- [22] Laura Vogelaar, Wietze Nijdam, Henk A.G.M. Wolferen van, René M. Ridder de, Frans B. Segerink, Eliane Flück, Laurens Kuipers, and Niek F. Hulst van. *Large area photonic crystal slabs for visible light with waveguiding defect structures: Fabrication with focused ion beam assisted laser interference lithography*. Advanced Materials, 13(20):1551–1554, 2001.

-
- [23] <http://www.epi-fab.eu/>.
- [24] Eiichi Kuramochi, Hideaki Taniyama, Takasumi Tanabe, Kohei Kawasaki, Young-Geun Roh, and Masaya Notomi. *Ultrahigh-Q one-dimensional photonic crystal nanocavities with modulated mode-gap barriers on SiO₂ claddings and on air claddings*. Optics express, 18(15):15859–15869, 2010.
- [25] Atsushi Uchida, Kazuya Amano, Masaki Inoue, Kunihito Hirano, Sunao Naito, Hiroyuki Someya, Isao Oowada, Takayuki Kurashige, Masaru Shiki, Shigeru Yoshimori, Kazuyuki Yoshimura, and Peter Davis. *Fast physical random bit generation with chaotic semiconductor lasers*. Nature Photonics, 2(12):728–732, DEC 2008.
- [26] A. Argyris, M. Hamacher, K. E. Chlouverakis, A. Bogris, and D. Syvridis. *Photonic integrated device for chaos applications in communications*. Physical Review Letters, 100(19), MAY 16 2008.

4

Caphe: A framework for simulating large networks of optical components

A large part of this PhD is devoted to the numerical modeling of optical circuits. However, the software that is present today does not meet the stringent requirements we need in the context of reservoir computing. For this reason, we have developed our own circuit simulator, called Caphe.

Caphe allows us to define building blocks which are coupled to other building blocks using ports. Each block can have an arbitrary number of ports and each component has its own set of arbitrary Ordinary Differential Equations (ODE) equations. This degree of freedom enables fundamental research on a variety of non-trivial components and circuits, with a quasi unlimited freedom in the way to define a topology, and the way to define and couple state variables.

Currently, the tool is used in numerous applications such as frequency-domain analysis of optical ring filters, time-domain analysis of optical amplifiers, microdisks and microcavities, and of course simulation of nanophotonic reservoirs. Although the original purpose of the simulator was to simulate optical circuits, the ODE equations do not limit us to optical components: different neuron types, and even financial models based on ODE equations, can be plugged into the simulator, which makes Caphe a very flexible and powerful tool for researchers in other domains as well.

This chapter is structured as follows: in section 4.1 we give a broad overview

of the simulation methods which are often used in nanophotonics. In section 4.2 we explain the concept of a scatter matrix, and propose a generic component which includes a scatter matrix, state variables and Ordinary Differential Equations (ODEs). In section 4.3 we create a circuit consisting of several nodes, and we explain how we can derive a generalized connection matrix from this circuit by eliminating linear, instantaneous nodes from the circuit. Then we describe how the time-domain integration routine works. In section 4.4, we show how we can speed up the calculations in frequency domain. Then we explain how we can use the generalized connection matrix (by solving the system in the frequency-domain), to speed up simulations in the time-domain considerably. In section 4.5 we give a few examples that demonstrate the usefulness of Caphe outside the domain of reservoir computing. We show a typical use-case of designing a filter in the frequency domain. Another example shows the time-domain simulation of a microring resonator using several dynamical variables such as the complex amplitude, temperature and free carriers. In section 4.6, we show how we use this framework to construct a nanophotonic reservoir. Then, in section 4.7 we explain why current software was not appropriate for our needs and which considerations were taken into account before starting the development of this simulator. Finally we conclude in section 4.8.

4.1 Numerical modeling

Analytical methods are useful to understand the underlying physics of an optical component. Furthermore they help us determine which are the relevant parameters when designing an optical component. For example, a higher refractive index usually means a smaller mode volume, or a smaller gap between two waveguides usually means a stronger coupling coefficient. However, once a design becomes too complex, it is no longer possible to solve Maxwell's equations analytically. The first thing to try in this situation is to approximate the behavior, in order to fall back on analytical methods. Sometimes this is not possible: the approximations can be too crude leading to wrong results, or some material properties are based on measured data. In these cases, numerical methods are an important tool to further design and/or optimize an optical component or circuit. There are a variety of tools available to simulate the behavior of light, using different techniques and different levels of approximations of Maxwell's equations. Many of these tools are limited to small networks of only a handful of components.

In order to illustrate several of the commonly used methods we will show how a light splitter is modeled in practice (see Figure 4.1). The light splitter, shown schematically in Figure 4.1 (right, bottom), has one port on the left and two ports on the right. In between there is a very wide waveguide that allows

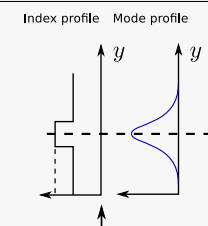
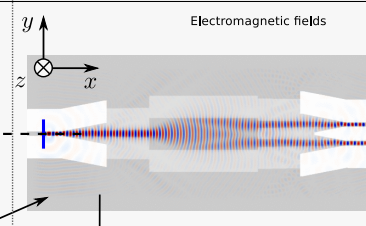
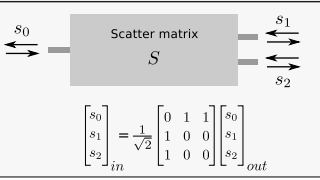
Simulation type	Eigenmode solver	FDTD
Software (dimensionality)	CAMFR (1D) FimmWave (2D)	Meep (2D) Meep (3D)
Example		
Simulation type	Custom scripts	Circuit simulation
Software (dimensionality)	Analytical or numerical model of a (sub)component (depends on the used model)	Caphe (linear with the number of components)
Example	<pre> def T1(wavelength): # Simple model: # half of the light up return sqrt(0.5) def T2(wavelength): # Half of the Light down return sqrt(0.5) s_1_out = T1(wavelength)*s_0_in s_2_out = T2(wavelength)*s_0_in </pre> <p>→ : Export to</p>	

Figure 4.1: Illustration of several simulation tools when designing a multi-mode interferometer (MMI). An eigenmode solver (top left) calculates the mode profile of a waveguide. This eigenmode is then used as input for a Finite Difference Time Domain (FDTD) simulation (top right). The output of this simulation can be sent to a circuit simulation tool. Also, users might want to perform a part of the simulation using their own code and link these to other tools.

multiple modes. Because the different modes interfere with each other, two nicely separated modes can be captured in the output ports by choosing an appropriate length of the multimode section. This device is called a multimode interferometer (MMI). The different simulation methods are explained below.

One of the most general ways to solve the Maxwell's equations numerically is the Finite Difference Time Domain (FDTD) method (a snapshot of the time evolution is shown in the top right of Figure 4.1). Here, Maxwell's equations are discretized in space and time. The smaller the discretization step, the more accurate the result will be but the longer it will take for the simulation to finish. The FDTD method is a time-domain method, and the simulations can be performed in 1, 2 or 3 dimensions. Although it can simulate about any geometry, with virtually no assumptions on the material properties, it is also the method that is computationally the most demanding. Especially in three dimensions and for resonant structures (in which light bounces back and forth at material boundaries), a single simulation can take hours to finish, and takes up a large amount of computer memory. The software package MEEP is an example of an FDTD simulation method, and is freely available as open-source [1, 2].

In many design problems the mode profile of a waveguide has to be known. If we take a cross-section of the waveguide, we can use an eigenmode solver to calculate the guided modes. A 1D cross-section and mode profile are shown in Figure 4.1 (top left). This type of numerical method works in the frequency domain. Because we only calculate a cross-section of the entire device (so we end up with a 1D or 2D geometry), a calculation over a reasonable frequency range usually takes a few minutes. Examples of eigenmode solvers are CAMFR [3, 4] and FimmWave [5].

When we consider the optical component as a black box, we do not discretize the component into smaller pieces. A passive and linear element, such as the MMI of Figure 4.1, is then reduced to a scatter matrix \mathbf{S} . Later in this chapter, we will see how we can link several optical building blocks together to build a complex circuit. Also, we will further enrich this building block by allowing nonlinear, non-passive behavior. With these additions, it becomes possible to simulate nonlinear circuits, such as a nanophotonic reservoir.

Furthermore, somewhere in between this spectrum of methods are the Time Domain Traveling Wave (TDTW) [6], the Split Step Methods (SSM) [6], and the Coupled Mode Theory (CMT, see section 3.2.1).

To summarize, of the listed simulation methods, the biggest difference lies in the complexity and the level to which they contain physical details. In its most general form, FDTD is a full-vectorial 3D optical solver. Although FDTD can cope with very complex geometries, it is computationally very expensive. On the other side, CMT is an approximate description, but extremely elegant and fast, only needing a few variables to describe a complex system. This is

achieved by eliminating all spatial dependencies in the physical problem.

In this software landscape, there are tools to design complex optical circuits consisting of many components. For example, ASPIC [7] is used for calculating the steady-state response of optical circuits, and VPI [8] is mainly used to study the time-domain evolution. PicWAVE uses a time-domain travelling wave (TDTW) optical model [9], and RSoft Optsim uses SSM [10]. There are also approaches that use Modified Node Analysis (MNA), such as OptiSPICE [11, 12], which allows simulation of mixed electrical and optical circuits. All of these new tools will become indispensable in the future when designing and optimizing large optical circuits.

In this chapter, we present a different node-based approach. The advantage of our approach is that both time and frequency domain can be investigated in the same framework, and that each component can be represented in a natural way using variables such as the optical field, the temperature and the carrier density, without needing to be mapped on to voltage or current such as in the SPICE-related MNA approach. It uses only a small set of variables per component, similar to CMT, which means the simulations are extremely fast compared to other methods such as FDTD, TDTW and SSM, with however the drawback of losing accuracy. Also, we provide a mechanism to eliminate instantaneous components from the network, reducing the number of components we need to simulate in the time domain. Our tool, named CAPHE [13] can therefore be used to efficiently simulate novel computational systems such as photonic reservoirs [14]. It is written in C++ for optimal performance, with a Python front-end for ease of use and interfacing to a large collection of scientific libraries.

4.2 Model

4.2.1 Scatter matrices

In this chapter we treat an optical structure as a black box which exchanges energy with the outside world through several physical outlets. Figure 4.2 shows how a structure has different physical outlets, which we will call optical ports (or ports, in short). These ports can be associated with an optical waveguide mode, or a free space electromagnetic beam. We assume that each port carries only one mode. This is not a restriction as different ports may physically coincide to describe waveguides with multiple modes.

We define a_i and b_i as the complex amplitude of the ingoing, resp. outgoing normalized electromagnetic mode. If the circuit is linear, we can define the following relationship between the outputs $\mathbf{B} = (b_0, \dots, b_{N-1})$ and the inputs $\mathbf{A} = (a_0, \dots, a_{N-1})$:

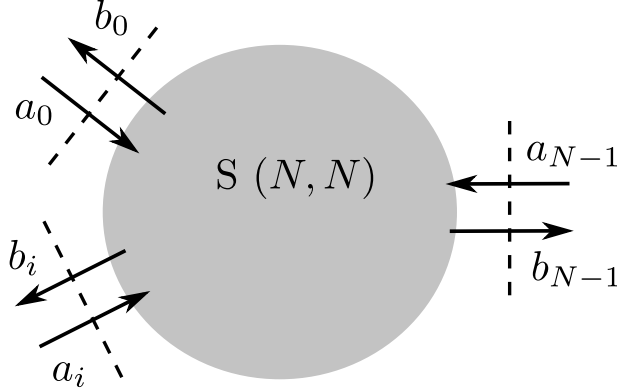


Figure 4.2: An N -port optical component which is treated as a black box. If the optical component is linear, the input-output relationship is fully determined by the scatter matrix \mathbf{S} .

$$\mathbf{B} = \mathbf{S} \cdot \mathbf{A} \quad (4.1)$$

Where we have defined the *scatter matrix* \mathbf{S} ($N \times N$). Depending on the actual material properties and geometry inside the structure, the component can have two interesting properties:

1. **Passive component:** A component is denoted passive when it is unable to generate energy. This means that (we assume the \mathbf{S} -matrix is time-invariant) $\|\mathbf{B}\| \leq \|\mathbf{A}\|$. If we square both sides and use $\|\mathbf{A}\|^2 = \mathbf{A}^H \mathbf{A}$, we get:

$$\mathbf{A}^H (\mathbf{I} - \mathbf{S}^H \mathbf{S}) \mathbf{A} \geq \mathbf{0}, \quad (4.2)$$

for all possible values of \mathbf{A} . This is equivalent to saying that the matrix $\mathbf{I} - \mathbf{S}^H \mathbf{S}$ is semi-positive definite.

For a lossless component, the equality holds. IN this case, this is equivalent to the condition that \mathbf{S} is an unitary matrix, i.e.,

$$\mathbf{S}^H \mathbf{S} = \mathbf{1}, \quad (4.3)$$

where \mathbf{S}^H is the conjugate transpose of \mathbf{S} .

When the \mathbf{S} -matrix is varying through time, the conditions for being passive are more complicated. Apart from the condition that the component should absorb more energy than it generates, the possible generation should happen *after* absorption. For more details we refer to [15].

2. Reciprocal: If a circuit is made of symmetrical constitutive parameters, i.e. the permittivity $\epsilon=\epsilon^T$ and the permeability $\mu=\mu^T$, then the circuit is called reciprocal. This is almost always the case, except for magnetic materials in the presence of a magnetic field. It can be proven that for this circuit, \mathbf{S} is a symmetrical matrix,

$$\mathbf{S} = \mathbf{S}^T, \quad (4.4)$$

and hence the transmission between port j to port k does not depend on the propagation direction.

Example: waveguide A waveguide with zero reflection can be represented by the following S-matrix:

$$\mathbf{S}_{wg} = \begin{bmatrix} 0 & A(\lambda) \exp(-j \frac{2\pi}{\lambda} L n_{eff}(\lambda)) \\ A(\lambda) \exp(-j \frac{2\pi}{\lambda} L n_{eff}(\lambda)) & 0 \end{bmatrix}, \quad (4.5)$$

where the effective index $n_{eff}(\lambda)$ can be calculated using an eigenmode solver. Here, λ is the wavelength of the light in vacuum, which is related to the frequency f by $\lambda f = c$. The loss in the waveguide is defined by the term $A(\lambda)$, and is usually wavelength dependent.

Example: directional coupler We define the directional coupler by the following scatter matrix:

$$\mathbf{S}_{DC} = \begin{bmatrix} 0 & 0 & \tau & j\kappa \\ 0 & 0 & j\kappa & \tau \\ \tau & j\kappa & 0 & 0 \\ j\kappa & \tau & 0 & 0 \end{bmatrix} \quad (4.6)$$

Where τ and κ are real numbers. Normally, the behavior of the directional coupler is determined by the geometry of the two waveguides and the gap between the two waveguides. Note that we have made abstraction of the geometry of the device just like we did with the waveguide: the phenomenological parameters τ and κ describe only the bulk behavior of this component. Reasoning on a circuit based on the τ and κ parameters is easier and more comprehensive compared to reasoning in terms of the actual geometry and gap between the waveguides. A relationship between the phenomenological parameters and the geometric parameters can be revealed by studying the actual physics of the device, for example by performing an FDTD simulation or by using an eigenmode solver. A directional coupler is lossless when $|\tau|^2 + |\kappa|^2 = 1$.

In the next section we extend this simple and effective description, such that we can model components that generate optical power (i.e. nonpassive), and contain nonlinearities.

4.2.2 Extension for S-matrices

The scatter matrix is a very elegant description of an optical component, however its functionality is rather limited. Here we extend the optical component—which from now on we will call a node¹— with different state variables and differential equations for these states. For example, the evolution of temperature and carrier density can be modeled in this way, and we can also incorporate the CMT equations we encountered in 3.2.1 to model a photonic crystal cavity. Furthermore, each node has access to its input history, which allows us to create delay lines or digital filters. Additionally, a node can contain subnodes, allowing the creation of hierarchical networks.

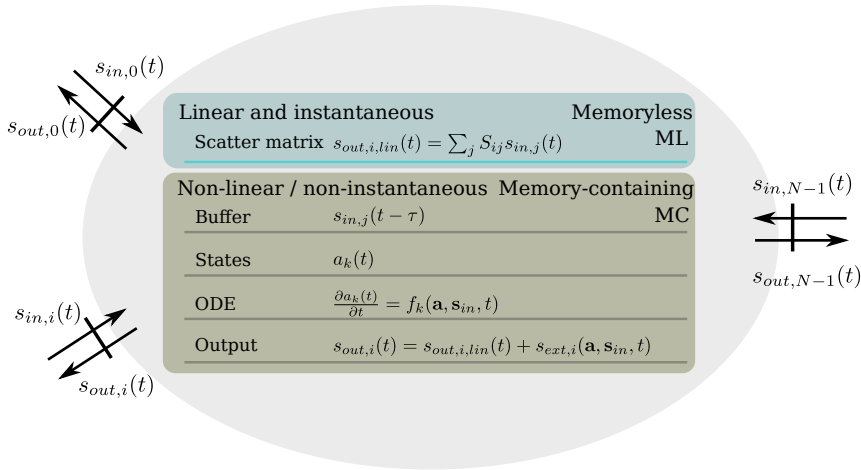


Figure 4.3: Structure of a node with N ports. A linear and instantaneous node is described by a scatter matrix \mathbf{S} . State variables (e.g. temperature and free carriers) can be added, accompanied by ordinary differential equations (ODE). In this case the node becomes non-instantaneous and can contain nonlinear behavior.

In Figure 4.3 we illustrate how one node is represented in Caphe. A node consists of N ports. A linear instantaneous transmission between port $s_{in,i}$ and $s_{out,j}$ is defined through the scatter matrix \mathbf{S}_{ij} .

Two optional time-domain descriptions can be added to enrich this component (see Figure 4.3, bottom): first, one can add a *buffer* to store the inputs $s_{in,i}$ at previous timesteps. This can be used if one wishes to model a delayed wave-

¹Not to be confused with a Node in the Oger toolbox. In the Oger toolbox (which depends on the Modular toolkit for Data Processing, MDP), a `mdp.Node` represents a block which can be executed and/or trained, such as the reservoir and the readout layer. In our context, the `mdp.Node` reservoir contains a collection of `caphe.Nodes`.

guide or a digital filter. Second, we can add internal *states* to the node. This can be used to describe for example the different population levels of a laser or the complex amplitude of a resonator. We use a set of ODE equations to describe the component in terms of its internal variables and input it receives. There is no restriction on the form of the equations, so we can also include nonlinear terms. Because of these two additions, the *output* $s_{out,i}$ is now a sum of the linear part and a term describing the temporal, possibly nonlinear behavior of the component.

The main novelty of our framework—compared to other optical simulation tools—is that the **S**-matrix of components with a fixed linear instantaneous transmission can be used to significantly speed up time-domain simulations of networks with both instantaneous and non-instantaneous components. In other words, as long as a component has no dispersion effects, it can be eliminated from the circuit. This is explained in section 4.3. This is mainly relevant in very large optical circuits, such as our nanophotonic neural networks, where many optical splitters would slow down the simulation.

The framework is also suited very well for simulating coupled mode theory equations, because it natively supports coupling matrices that couple the input to the states, the states to the output, and the modes to themselves (this is also described in more detail in section 5.4.1, where we calculate the Jacobian of a system of CMT equations). This allows us to calculate the steady-state transmission of linear CMT equations out-of-the-box.

4.2.3 Carrier modulation

We represent time-domain signals as complex amplitudes $s(t)$. The actual field is then the product of the very fast carrier (with frequency ω_c), modulated by a complex-valued envelope $s(t)$:

$$E(t) = s(t)e^{j\omega_c t} + c.c. \quad (4.7)$$

Representing the signal by the *envelope* $s(t)$ instead of $E(t)$ is beneficial from a numerical point of view, because $s(t)$ varies much slower than $E(t)$, hence we can choose a much larger integration step. Obviously, as the bandwidth of the input signal $s(t)$ increases, we will need more samples per time unit to accurately simulate the system. If the input consists of K signals $s_k(t)$ on clearly separated frequency bands $\omega_{c,k}$, $k \in [0..K-1]$, then a sum of multiple signals, each with its own carrier frequency, can be used. In this case, the actual field is given by:

$$E(t) = \sum_{k=0}^{K-1} s_k(t)e^{j\omega_{c,k}t} + c.c. \quad (4.8)$$

4.2.3.1 Comparison with circuit envelope simulation

Also in high-speed electronics, it can be beneficial to represent a signal as a fast carrier (with frequency f_c) modulated with a slow-varying envelope (with highest frequency f_s), where $f_s \ll f_c$. In case there is no nonlinearity in the circuit that causes harmonics of the carrier wave to appear, it is relatively easy to create a SPICE-compatible equivalent circuit. The advantage is that the timestep only has to be small enough to capture the bandwidth of the modulation envelope (f_s) [16, 17]. The envelope simulation is thereby much faster than the full cycle-by-cycle simulation of the original circuit. The way to represent these time-varying signals is equivalent to the solution proposed above (see equation (4.7)). The difference is that in electronics, underneath is a SPICE model, whereas in the photonics case, we use the framework which is proposed in this chapter. As we already explained in the introduction of this chapter, it is beneficial to represent signals in a natural way: for electronics, using current and voltage, and for optics, using the optical field. The main reason for this is that optical fields are non-conservative quantities, and therefore cannot be represented as voltages and currents [11].

Most of the time, one has to take into account nonlinear effects in electronic devices, which cause harmonics to be generated at $2f_c$, $3f_c$ and so on. In this case, a harmonic balance simulation can be used to find the energy distribution of the fundamental frequency and its harmonics. This technique works iteratively, whereby the simulation is split into a linear part, which is solved in the frequency domain, and a nonlinear part, which is solved in the time-domain. The simulation is converged when Kirchoff's circuit laws are obeyed.

The concept of using modulated signals in combination with harmonic balance simulations, is called the circuit envelope simulation, or the Envelope Transient Harmonic Balance technique (a good overview of the available methods is given in the introduction part of [18]). The time-evolution can be performed with large timesteps (linked to f_s), and for each timestep a harmonic balance simulation is performed to find the power distribution in the relevant harmonics. Even though at each timestep a harmonic balance simulation is performed, it is still faster than simulating cycle-per-cycle with the much higher frequency f_c .

In conclusion, the principles of separating two different timescales (the fast carrier and slow envelope) is equivalent in the proposed software framework and the circuit envelope simulation. However, the proposed framework here does not yet take into account mixing of different carrier frequencies. One could think of an optical equivalent of the harmonic balance technique, although it

would not be used to calculate harmonics of the carrier², but for example to calculate the mixing terms of a four-wave mixing process (see for example the coupled mode equations in [19], and [6]).

4.2.4 Generalized source term

For each component we can now optionally add time-domain equations which leads in its most general form to an input-output relation of the following form:

$$s_{out,i}(t) = \sum_{j=0}^{N-1} \mathbf{S}_{ij} s_{in,j}(t) + s_{ext,i}(\mathbf{a}, \mathbf{s}_{in}, t) \quad (4.9)$$

$$s_{out,i}(t) = s_{out,i,lin}(t) + s_{ext,i}(\mathbf{a}, \mathbf{s}_{in}, t) \quad (4.10)$$

Here, $s_{ext,i}$ is a generalized source term. E.g. for a continuous wave source, $s_{ext} = A$, where A is the complex amplitude of the source. For a two-port waveguide with delay τ , the simple relation $s_{out,i}(t) = s_{ext,i}(t) = B s_{in,1-i}(t - \tau)$, for $i \in [0, 1]$, holds. Here, B is the complex value which determines the loss and phase change of the waveguide. Note that for this waveguide, there is no longer an instantaneous behavior, i.e. \mathbf{S}_{ij} is zero in (4.9)³.

Also note that this description does not take into account waveguide dispersion: both the transmission B and the delay τ are fixed and calculated for the carrier frequency ω_c . By using a digital filter, of which the output is stored in the generalized source term, it is possible to incorporate dispersion effects

As soon as there is a source term present in (4.9), the component is not instantaneous anymore. We call these nodes memory-containing (MC) nodes (Figure 4.3, bottom), as opposed to the memoryless (ML) nodes. Depending on whether the delays in a waveguide are important for a simulation, one can model them with delay (which makes it MC), or without delay (as a ML component), the latter having the advantage that we can eliminate it from the total network. This is explained in the next section.

4.3 Towards a circuit

4.3.1 Generalized connection matrix

We use the node from Figure 4.3 as a basic building block to create an optical circuit. An optical circuit consists of several nodes, of which the ports are linked

²Silicon does not have a second-order nonlinearity due to the symmetry of the crystal. The third-order nonlinearity is also very weak, so one would need very high powers to generate harmonics. For telecom wavelengths, the generated harmonics would also be absorbed by the silicon.

³The output of the waveguide is $s_{out,j}(t) = \mathbf{S}_{ij} s_{in,1-j}(t) + A s_{in,1-j}(t - \tau)$. If we would use a non-zero \mathbf{S} matrix, we would count the same input twice, which is clearly wrong.

to the ports of other nodes in the circuit as we will explain in this section. An important matrix in our approach is a matrix which we will call the generalized connection matrix, which describes how all inputs s_{in} are related to the generalized source term s_{ext} . This is done by eliminating the ML nodes from the circuit. This is a crucial feature in our approach, and has two direct results. The first outcome is that we have now calculated the frequency-domain response of the system for all MC nodes. Second, in the time domain, it reduces the number of variables needed, hence improving the simulation speed.

To perform the elimination of ML nodes, we split the input/output vector $\mathbf{s}_{in/out}(t)$ into $\mathbf{s}_{in/out,MC}(t)$ and $\mathbf{s}_{in/out,ML}(t)$, for MC and ML nodes. For simplicity we will omit the time dependency in the following equations.

We can describe the way the different components are connected as follows:

$$\begin{pmatrix} \mathbf{s}_{in,MC} \\ \mathbf{s}_{in,ML} \end{pmatrix} = \mathbf{C}_{tot} \begin{pmatrix} \mathbf{s}_{out,MC} \\ \mathbf{s}_{out,ML} \end{pmatrix} = \begin{pmatrix} \mathbf{C}_{MC,MC} & \mathbf{C}_{MC,ML} \\ \mathbf{C}_{ML,MC} & \mathbf{C}_{ML,ML} \end{pmatrix} \begin{pmatrix} \mathbf{s}_{out,MC} \\ \mathbf{s}_{out,ML} \end{pmatrix}. \quad (4.11)$$

Here, $\mathbf{C}_{tot,ij}$ is a binary connection matrix which only contains a 1 if port i is connected to port j . As a consequence, \mathbf{C}_{tot} is symmetric and contains at most 1 element per row and at most 1 element per column, with zeros on the diagonal.

The behavior of each of the individual nodes can be described by the following equations:

$$\mathbf{s}_{out,ML} = \mathbf{S}_{ML,ML} \mathbf{s}_{in,ML} \quad (4.12)$$

$$\mathbf{s}_{out,MC} = \mathbf{S}_{MC,MC} \mathbf{s}_{in,MC} + \mathbf{s}_{ext,MC}, \quad (4.13)$$

in which we define the scatter matrices $\mathbf{S}_{ML,ML}$ and $\mathbf{S}_{MC,MC}$. These are block diagonal matrices, with each block representing the scatter matrix from a ML resp. MC node. The second term in equation (4.13), $\mathbf{s}_{ext,MC}$, is the generalized source term described earlier, see also equation (4.9).

Using all the equations above we can derive the input at the active ports, given only $\mathbf{s}_{ext,MC}$. This is done as follows: replace $\mathbf{s}_{out,ML}$ in equation (4.11) using (4.12), then solve this system for $\mathbf{s}_{in,MC}$. This gives

$$\mathbf{s}_{in,MC} = \left(\mathbf{C}_{MC,MC} + \mathbf{C}_{MC,ML} \mathbf{S}_{ML,ML} (\mathbf{I} - \mathbf{C}_{ML,ML} \mathbf{S}_{ML,ML})^{-1} \mathbf{C}_{ML,MC} \right) \mathbf{s}_{out,MC} \quad (4.14)$$

$$= \mathbf{C} \mathbf{s}_{out,MC} \quad (4.15)$$

We then substitute (4.13) in the equation above. This yields

$$\mathbf{s}_{in,MC} = (\mathbf{I} - \mathbf{C} \mathbf{S}_{MC,MC})^{-1} \mathbf{C} \mathbf{s}_{ext,MC} \quad (4.16)$$

$$= \mathbf{C}_{exttoin} \mathbf{s}_{ext,MC} \quad (4.17)$$

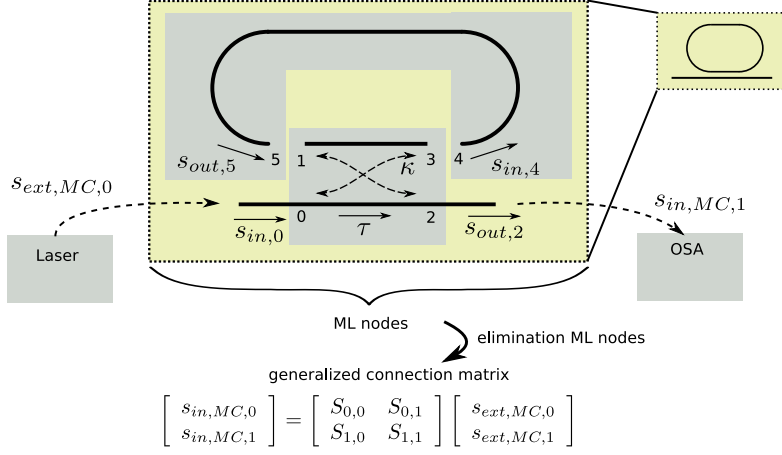


Figure 4.4: Illustration of a microring resonator. It consists of two parts: the directional coupler and the (bent) waveguide. We also show the two memory-containing ports, which come from a laser and optical spectrum analyzer (OSA). All memoryless nodes are eliminated from the circuit, so we end up with a small (2x2) generalized connection matrix S of the circuit.

We have now successfully eliminated the memoryless nodes and end up with a smaller matrix $\mathbf{C}_{exttoin}$ (the generalized connection matrix) of the network. The notation *exttoin* is used to illustrate that it transforms the generalized source term \mathbf{s}_{ext} to the inputs \mathbf{s}_{in} .

Note that the matrix inversion in (4.14) is of a special type: $\mathbf{C}_{ML,ML}$ only permutes the elements of $\mathbf{S}_{ML,ML}$, and $\mathbf{S}_{ML,ML}$ is a block diagonal matrix. The resulting matrix $\mathbf{I} - \mathbf{C}_{ML,ML}\mathbf{S}_{ML,ML}$ is therefore sparse. However, the resulting matrix after inversion is not always sparse. This depends on the topology of the original network and on the individual scatter matrices of the ML nodes.

Example: microring resonator. We demonstrate the process of node elimination with a simple example. Consider the microring resonator of Figure 4.4, which is the combination of a directional coupler and a bent waveguide (we have defined the S matrices of these components in section 4.2.1). The bent waveguide connects the two top ports of the directional coupler. We have two MC nodes in the circuit, a laser and an Optical Spectrum Analyzer (OSA). We can solve this system using the formulas (4.14) and (4.16) as we just explained, but in this case it is easier and more instructive to solve the system immediately without creating all intermediary matrices. In the microring resonator, monochromatic light (having a frequency ω_c) circulates inside the ring, and the

output is an infinite sum of field contributions with decreasing amplitude, of the form

$$s_{out,3} = \kappa s_{in,0} (1 + a + a^2 + \dots), \quad (4.18)$$

where $a = A\tau \exp(-j\beta L)$, and $\beta = \frac{2\pi}{\lambda} n_{eff}(\lambda)$. We assume that the loss A is independent of the wavelength, which is a reasonable assumption for the bandwidth that we consider. This infinite sum only exists when $|a| < 1$, and equals:

$$s_{out,3} = \kappa s_{in,0} \frac{1}{1 - A\tau \exp(-j\beta L)}, \quad (4.19)$$

which leads us to the the well-known equation for a microring resonator (where it is assumed that the directional coupler is lossless, i.e. $|\tau|^2 + |\kappa|^2 = 1$):

$$s_{in,MC,1} = s_{ext,MC,0} \frac{\tau - A \exp(j\beta L)}{1 - A\tau \exp(j\beta L)}. \quad (4.20)$$

Note that if we followed the derivation of equations (4.14) and (4.16), we obtain the same solution, namely $\mathbf{C}_{exttoin,1,0}$. The case $|a| > 1$ will not occur, because this would mean $\tau > 1$. In the case of $a = 1$, the determinant of $\mathbf{I} - \mathbf{C}_{ML,ML} \mathbf{S}_{ML,ML}$ becomes zero. This will only occur in the pathological case where $\tau = 1$, which describes a directional coupler without coupling, and when the ring is at resonance, i.e. $\beta L = 2\pi$. Even then, in practice this is not a problem as in the case where $\tau = 1$, the undefined fields (associated to ports 1,3,4 and 5) do not appear in the generalized connection matrix. The result in the case when $\tau = 1$ is unit transmission: $s_{in,MC,1} = s_{ext,MC,0}$.

If the bandwidth of the input signal is small compared to the bandwidth of the ring resonator⁴, it is safe to eliminate the ML nodes before starting the time-domain simulations. Instead of using 8 ports, we would then only use 2 ports. If the signal bandwidth is not negligible, first a digital filter has to be constructed to account for the wavelength dependent $\mathbf{C}_{exttoin}$.

4.3.2 Integration in time-domain

The final ingredient in our model are the internal states, which are stored in the total variable vector $\mathbf{a}(t)$. These describe the evolution of some internal variables, e.g., temperature and free carriers in a laser, as a function of time and inputs:

$$\frac{d\mathbf{a}(t)}{dt} = \mathbf{f}(\mathbf{a}, \mathbf{s}_{in}, t) \quad (4.21)$$

⁴For a ring with a Q-factor of 4500, the 3 dB bandwidth (if the ring has a resonance around 1550 nm) is 0.34 nm, which corresponds to a signal bandwidth of 42 GHz. The phase change over the 3dB bandwidth is $\pi/2$ radians, which is not negligible.

In order to perform an integration, we have to evaluate the function \mathbf{f} . This is done as follows: at each timestep we loop over all MC nodes to calculate the generalized source term \mathbf{s}_{ext} . Then we can calculate the input to each MC node, \mathbf{s}_{in} , from equation (4.16). When we know all inputs, we can evaluate the ODE equations for each node, $f_k(\mathbf{a}, \mathbf{s}_{in}, t) = \frac{da_k(t)}{dt}$, and concatenate them to form $\frac{d\mathbf{a}(t)}{dt}$.

This derivative can then be used to drive an integration routine. For example, in the case of a forward Euler integration, a fixed timestep Δt is used, and the next state is given by:

$$\mathbf{a}(t + \Delta t) = \mathbf{a}(t) + \Delta t \mathbf{f}(\mathbf{a}, \mathbf{s}_{in}, t) \quad (4.22)$$

Two other integration routines are provided to the user. The first one is the classical Runge-Kutta method, which is a fourth-order method, meaning that the error per step is on the order of $O(\Delta t)^5$, while the total error accumulation is of the order of $O(\Delta t)^4$. Compared to the first-order Euler integration it is much more accurate, but it requires more function evaluations. The last integration routine is based on Bulirsch-Stoer [20]. Here, we combine this algorithm with an adaptive stepsize method, in which the internal integration step Δt is modified during the integration, where the integration step should be as large as possible while maintaining a given accuracy.

4.4 Optimizations

In this section we discuss how we can speed up the frequency and time domain calculations. First, we describe how we can significantly improve the scalability of the software by using sparse matrices, and we perform a test to check the speed improvements. Next, we demonstrate how the elimination of linear, passive nodes, leads to less memory requirements and lower simulation time.

4.4.1 Optimizations in the frequency domain

In equation (4.14) and (4.16) we need to solve a system of equations. For example, in equation (4.16) we solve

$$(\mathbf{I} - \mathbf{C}\mathbf{S}_{MC,MC})\mathbf{X} = \mathbf{C} \quad (4.23)$$

for \mathbf{X} . This can be done by first doing a LU factorization, followed by forward and backward substitution to find \mathbf{X} . A similar reasoning is done for the inversion in (4.14). Solving a system is almost always preferred above matrix inversion in terms of speed and stability. Optionally, since these matrices are sparse, we can use rely on sparse matrix algorithms to solve this system. One software package, the Clark Kent sparse LU factorization algorithm (KLU), is optimized for circuit-like matrices, and so it is well suited for our application [21–23].

To benchmark the speed gain, we consider the use case of an optical filter called a Coupled Resonator Optical Waveguide (CROW). A CROW is a sequence of optical rings as shown in Figure 4.5. Each section is made of a directional coupler (with coupling values κ_i) and two waveguides, which then couples to the next section. Here we will only perform a speed benchmark, in section 4.5.1 we consider a real-world use case.

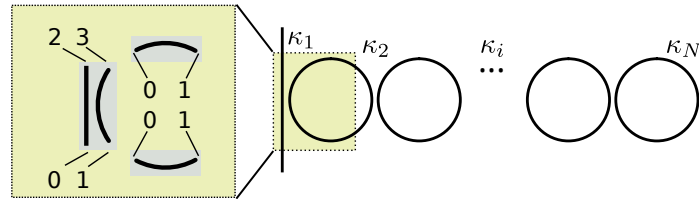


Figure 4.5: A Coupled Resonator Optical Waveguide (CROW). Each section is subdivided in a directional coupler and two waveguides. Port numbers are shown in the left.

The directional coupler and the waveguide are ML components with four resp. two ports. This means there are eight ports per CROW section. By plotting the simulation time as a function of the number of ports in the circuit, we observe how scalable our software framework is for the frequency domain. This is shown in Figure 4.6, where we compare the time spent by different matrix strategies as a function of the number of ML ports. As can be seen in the figure, a large number of CROW sections can easily be handled with the KLU algorithm. This proves the technique is useful for analyzing very complex systems in steady-state regime.

4.4.2 Improving the simulation speed in the time domain

As already mentioned, if a network contains both ML and MC nodes, one can eliminate the ML nodes prior to starting the time domain simulation. The speed of the time domain simulation depends on the size of the generalized scatter matrix after eliminating the ML nodes. To benchmark the speed improvement, we simulate a large network of components. The nanophotonic reservoirs, which are the subject of this dissertation, are perfectly suited for this benchmark task. More specifically, we simulate a nanophotonic reservoir consisting of Semiconductor Optical Amplifiers (SOA). Each SOA is connected to its nearest neighbours, in a structure called a swirl topology [24], see Figure 4.7(a). To connect the SOAs, we used a combination of splitters and waveguides. The actual creation of the circuit is considered in-depth in section 4.6.

We compare two systems. In the first system the ML nodes behave as MC nodes, in the second system we first eliminate all ML nodes. In the first case,

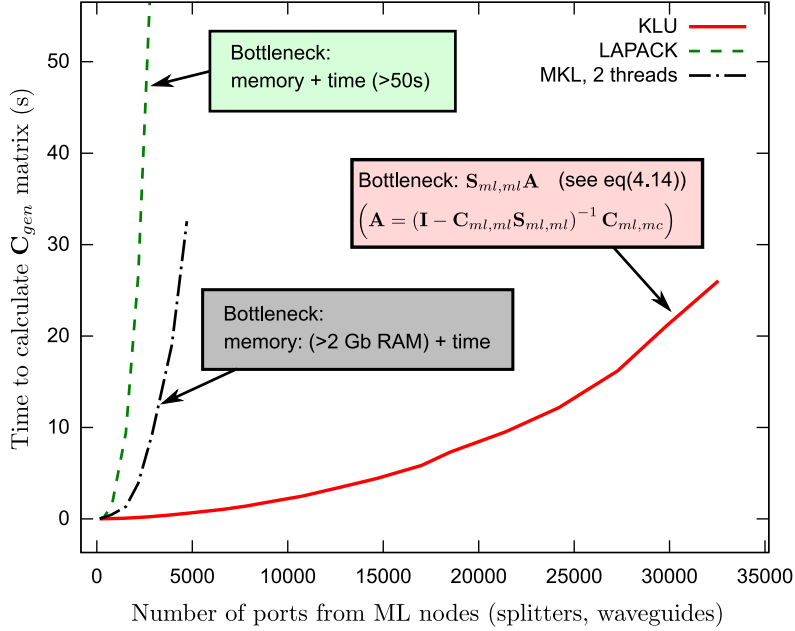


Figure 4.6: Calculating the frequency response of a passive network. Using KLU, a sparse matrix solver suited for circuit-like matrices, we can easily calculate scatter matrices of very large networks.

we need to calculate the light propagation in each splitter and waveguide separately, which means the simulation will take longer, and consumes more memory, as in the second case. This is illustrated in Figure 4.7(b).

The total simulation time is mainly determined by the evaluation of the ODEs of the individual SOAs and the matrix multiplication from equation (4.16). There is a clear benefit of eliminating the ML nodes: the simulation speed is approximately halved as shown in Figure 4.7(a) (top). The calculation time for evaluating the ODEs is the same for both systems. The memory usage is shown in the bottom graph of Figure 4.7(a): It is a sum of the memory allocated in C++ and in Python. The offset is due to initialization overhead in Python. For all simulations, we used a buffer that stores 500 timesteps. Because we eliminated the ML nodes, the memory requirements are greatly reduced. Since we use sparse matrices, calculation time and memory requirements scale linearly as a function of the network size.

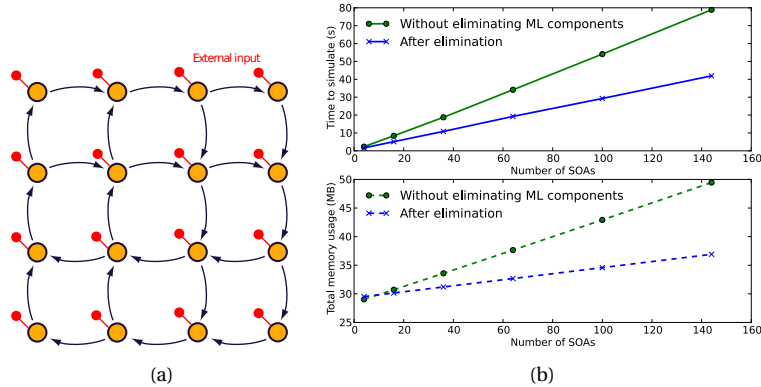


Figure 4.7: Left: topology used to simulate a complex system with ML and MC nodes. Each circle represents a SOA. Splitters are not shown. Right: the simulation time and memory usage increases linearly with the number of SOAs. Clearly there is an advantage by eliminating the ML nodes, both in terms of speed and memory usage.

4.5 Examples

The framework presented here is very well suited for calculating the transmissions and reflections in large networks, and to study the dynamics of nonlinear coupled systems. The first example shows how we design a Coupled Resonator Optical Waveguide (CROW) with a flat pass band. Although we can simulate very large networks (see also Figure 4.6), we will demonstrate the tool with a limited number of rings. Using scientific libraries which are readily available in Python, we can optimize this circuit very easily. In the second example we demonstrate the nonlinear dynamics of ring resonators in different topologies, under the influence of free carriers and temperature effects.

4.5.1 Coupled Resonator Optical Waveguide

Here we consider the Coupled Resonator Optical Waveguide (CROW) network as we showed in Figure 4.5, with four rings. By adjusting the coupling strengths κ_i , $i \in [1, 5]$, of the coupling sections, we can design optical filters with a desired shape, such as a flat band filter with a certain wavelength range, as shown in Figure 4.8.

Here we design a pass-band filter of approximately 1 nm around $1.545 \mu\text{m}$. This resonance wavelength is determined by the roundtrip length of one oscillator, which we will keep fixed. To find a set of κ_i , we will use an evolutionary algorithm. Since Caphe has a Python front-end, we have access to all Python

optimization libraries. In this case, we use the Covariance Matrix Adaptation Evolution Strategy, (CMA-ES) to optimize the coupling coefficients κ_i , $i \in [1, 5]$, where we assume $\kappa_1 = \kappa_5$ and $\kappa_2 = \kappa_4$. For more details about the algorithm, we refer to [25]. The penalty is determined by the mean square error between the resulting filter for a certain set of κ_i and the target filter. Each simulation takes about 200 milliseconds. After 33 generations with a population size of 14 (which takes a few minutes on a desktop computer), we get a solution that is very close to the desired function, see Figure 4.8. The coupling values we used were $\kappa = [0.285, 0.017, 0.009, 0.017, 0.285]$.

We stop the optimization routine here because further small improvements will not necessarily result in better results. This is mainly due to statistical variations in the geometry of the devices, which have influence on the κ values and effective indices of the waveguides. These variations are in fact detrimental for the performance of the filter. This can be seen in Figure 4.8(b), where we simulate different random process variations⁵, and it is clear that the shape of the filter is not optimal.

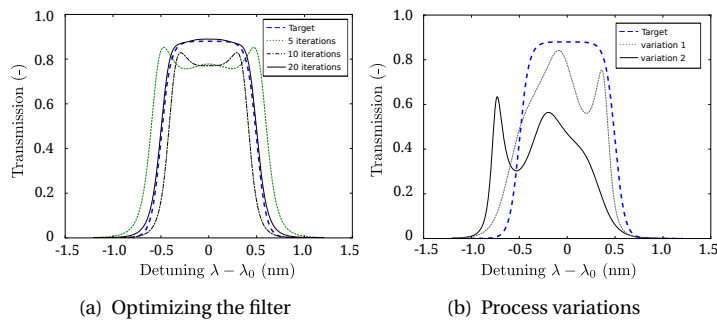


Figure 4.8: CROW: Optimizing the κ_i to match a certain filter (left). With process variations, performance deteriorates (right).

We can compensate for the process variations by changing the refractive index locally using micro-heaters on top of the waveguides. Suppose the ring resonances due to process variations can vary over 1 nm [26], then the refractive index change needed to compensate this variation is approximately 0.0022. Using $dn_{eff}/dT \approx 1.86 \cdot 10^{-4}$ [27], this is about 11.82°C. Using micro-heaters on top of the waveguides, we can achieve this temperature variation in a ring using less than 1 mW, depending on the exact design of the micro-heaters (see [28, 29], and references therein). If we assume that the resonance shifts due to process variations are uniformly distributed, we need approximately 6°C thermal tuning per ring, which corresponds to about 3 mW.

⁵In this case, a random variation on the resonance wavelength of 0.5 nm was used.

4.5.2 Dynamics of three coupled ring resonators in a feedback loop

High-Q cavities show interesting nonlinear dynamic behavior when excited in the correct regime. For instance, self-pulsation is experimentally observed in small, high-Q microrings, besides the expected thermal bistability [30]. This self-pulsation can be described phenomenologically, in a very accurate way, using Coupled Mode Theory (CMT) [31]. The microring is represented by four dynamical variables: two complex amplitudes representing the energy and phase of the light travelling in the CW and CCW direction, and two real variables, one representing the temperature difference between the ring and its environment, and a second representing the number of free carriers. This system thus inherently contains a lot of different timescales: the temperature time constant (approx. 100 ns - 1 μ s), the free carrier relaxation time (approx. 1-10 ns), the coupling between the ring and the bus waveguide (approx. 10-100 ps) and the coupling between the CW and CCW mode (which can also be faster than nanoseconds). Given the different timescales and the compact formulation of the basic equations, our tool is very well suited to simulate this system. In Figure 4.9 we show how different fixed input powers can trigger the experimentally observed self-pulsation in an all-pass filter. The exact formulation of the differential equations governing this system, and the order of magnitude of the appropriate constants needed to observe the self-pulsation, can be found in [30, 31].

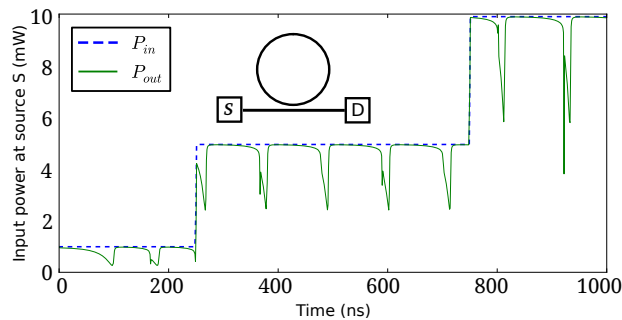


Figure 4.9: Self-pulsation in a single (all-pass) microring resonator.

However, the real use case of our tool is simulating photonic circuits containing more than one component. We demonstrate this in Figure 4.10 by investigating a system where we couple three self-pulsating microrings with an external feedback loop, with zero roundtrip phase at the signal wavelength. Two 3dB splitters connect the system with respectively a source and a detector.

In the systems of Figure 4.9 and Figure 4.10, self-pulsations are present, which arise from the interplay between the temperature and plasma dispersion

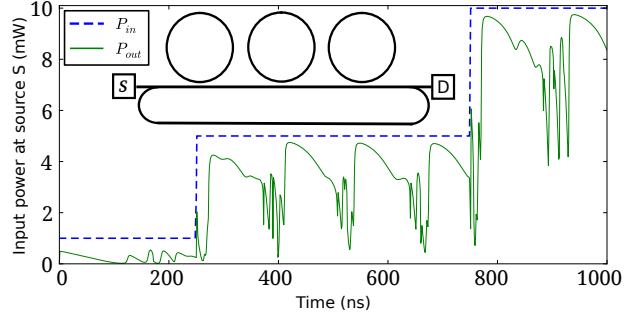


Figure 4.10: Dynamics of a system with three (all-pass) microring resonators coupled with a feedback loop (zero roundtrip phase at the signal wavelength), containing two 3dB-splitters, connecting the loop with resp. a source and a detector.

effect. The purpose is to investigate the dynamics in a single ring and in coupled rings, and to see how they could be used as an excitable structure. These rings could then be used as spiking neurons, in order to create a new type of artificial neural network. This research is continued by T. Van Vaerenbergh [32, 33].

4.6 Constructing a nanophotonic reservoir

To lay the groundwork for later chapters, we explain in this section how to construct a nanophotonic reservoir. First, we explain which constraints we have on the topology, and propose a mesh-like topology. Then, we show how this is implemented in our software framework.

4.6.1 Hardware topology

A hardware implementation of a reservoir implies certain restrictions on the topology. E.g., the chip is planar and waveguide crossings are to be avoided, which implies regular topologies as opposed to random topologies.

The neurons that make up the nanophotonic reservoir are usually two port components, such as the SOA, the PhCC and the microring resonator. This means we will need an optical splitter to distribute the light coming from these neurons to other neurons. For example, a 3-port splitter can distribute the output of one neuron to two other neurons. If we want the output to be distributed over three neurons, we need a 4-port splitter. And so on. However, it is not possible to create a lossless 3-port splitter, and it is not possible to create a lossless reciprocal 4-port splitter. Table 4.1 summarizes the options for $N=3$ to $N=6$

N	Lossless	Reciprocal
3	No	/
4	Yes (eq. (4.24))	No
5	Yes [34]	Yes
6	Yes [34]	Yes

Table 4.1: List of possible splitters with N ports, where the power is equally distributed over N-1 output ports. For some N, it is not possible to create a lossless splitter. Advanced magneto-optic materials can be used in order to break reciprocity on the SOI platform [35], but make the fabrication more complex.

ports. The S-matrix of a nonreciprocal lossless 4-splitter could look like [34]:

$$\mathbf{S}_{splitter} = \frac{1}{\sqrt{3}} \begin{bmatrix} 0 & 1 & 1 & -1 \\ 1 & 0 & 1 & 1 \\ -1 & 1 & 0 & 1 \\ -1 & -1 & 1 & 0 \end{bmatrix} \quad (4.24)$$

Because of the highly increased design complexity for splitters with more than three ports, and because they will more likely be susceptible to process variations, we will use the standard three port splitters which are well tested on the SOI platform, such as an MMI with one input arm and two output arms (for example the one shown in Figure 4.1), or a Y-junction. The scatter matrix for a 3 dB splitter (which splits power equally in two arms) is given by:

$$\mathbf{S} = \begin{bmatrix} 0 & \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} & 0 & 0 \\ \frac{1}{\sqrt{2}} & 0 & 0 \end{bmatrix} \quad (4.25)$$

Also, the structure needs to be planar and too many optical crossings should be avoided if possible because these cause loss and cross-talk. Recent improvements in crossing strategies reduce the loss to only 0.17 dB [36] per crossing, which means the restrictions on the topology are now less stringent.

In this dissertation we use two topologies. A *swirl* topology (see Figure 4.7(a)) and a *waterfall* topology (see Figure 4.11 and Figure 4.12). The swirl topology was proposed by K. Vandoorne [14, 24]. With this planar topology, special care was taken to mix the signals by adding a significant number of explicit feedback loops. This is necessary because the SOA is a unidirectional component, i.e. there is no reflection. On the other hand, photonic crystal cavities do show significant reflections, so even in the waterfall topology, feedback loops and signal mixing will occur. This is the topology that is used most of the time in this dissertation.

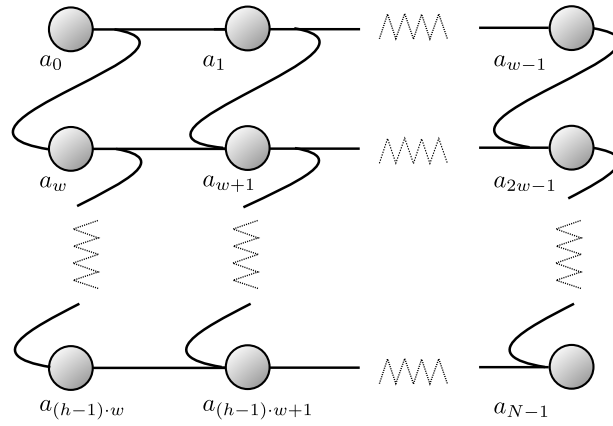


Figure 4.11: The proposed topology for the nanophotonic reservoir. Each circle represents a PhCC, and each splitter represents an MMI or Y-junction. Due to hardware restrictions, it is difficult to use a random topology. Especially the fan-in should be minimized, because a large fan-in means a higher sensitivity to process variations and increased design complexity. A regular mesh topology, such as the waterfall topology shown here, minimizes the fan-in and crossings, while keeping a good connectivity. A reservoir based on the waterfall topology has a good performance, can be easily designed and minimizes the amount of fan-in and fan-out.

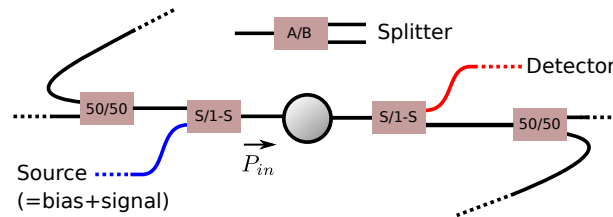


Figure 4.12: Illustration of the typical splitting ratios in a fully connected node in the waterfall topology (a 50/50 splitter is also called a 3dB splitter). The fan-in and fan-out have been minimized to three. An important design parameter is the splitting ratio S . It is the fraction of power that enters the network from the source, and the fraction of power that goes to the detector.

4.6.2 Implementation of the network

In order to create this network, we have made several intermediate building blocks. The most general class is called the `ConnMatrixNetwork`, and is used for almost all simulations in the following chapters. Based on a connection matrix \mathbf{C} and a set of building blocks $BB(k)$, it creates the actual circuit. During the process, we heavily rely on hierarchical features of our framework. We will see later on how flattening the circuit, i.e. removing all hierarchy, can improve the speed of the simulation.

4.6.2.1 Creating the circuit

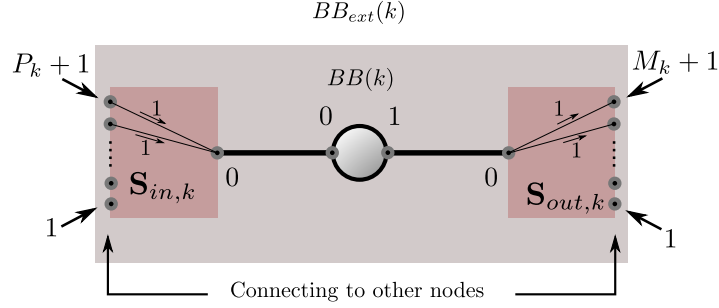
The class `ConnMatrixNetwork` is a complex class with many arguments. We will only highlight the parameters which are relevant to demonstrate the creation of the network (the other parameters can be found in the class documentation):

- Building blocks (BB): This is a list of basic building blocks that make up the circuit (size N). We have no restriction on the choice of building blocks. This means we are not bound to one neuron type, so it is possible to combine different optical building blocks (e.g. combining side-coupled cavities with inline coupled cavities).
- Connection matrix (\mathbf{C}). If $\mathbf{C}(k, l)$ is different from zero, then there is a connection from $BB(l)$ to $BB(k)$ with strength $\mathbf{C}_{k,l}$.

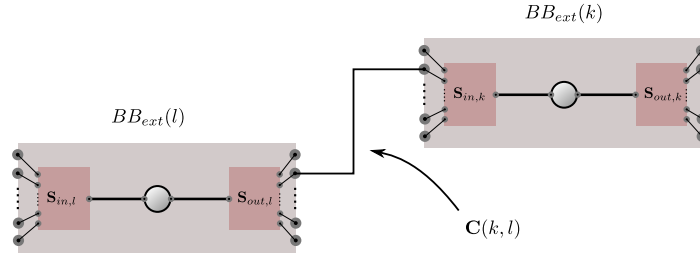
Creating the network is done in two steps: first we extend each building block with splitters, as shown in Figure 4.13(a), which are then called extended building blocks, BB_{ext} . The input and output splitters have a scatter matrix $\mathbf{S}_{in,k}$ and $\mathbf{S}_{out,k}$, respectively. If $BB(k)$ is linked to M_k other elements, then $\mathbf{S}_{out,k}$ is a $(M_k + 1, M_k + 1)$ matrix (1 input on the left, M_k outputs on the right). Also, for P_k inputs on $BB(k)$, $\mathbf{S}_{in,k}$ is a $(P_k + 1, P_k + 1)$ matrix. The $\mathbf{S}_{in,k}$ matrix has the following structure:

$$\mathbf{S}_{in,k} = \begin{bmatrix} 0 & \overbrace{1 \cdots 1}^{P_k} \\ 0 & 0 \quad 0 \\ \vdots & \ddots \\ 0 & \cdots 0 \end{bmatrix}$$

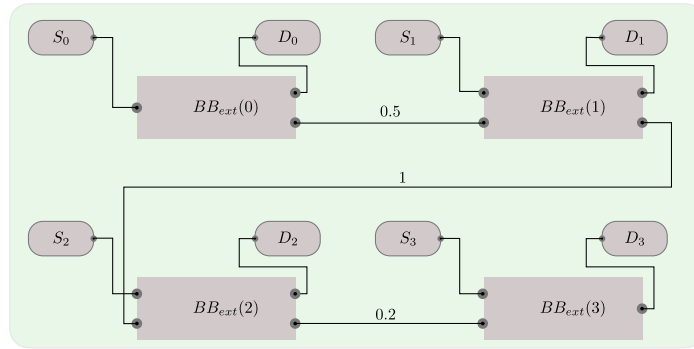
$\mathbf{S}_{in,k}$ contains only 0/1 values, and port 0 is the input for $BB(k)$. Because the splitter has 1 values, the actual power splitting should be taken care of by the \mathbf{C} matrix. This is done on purpose, such that we can *probe* the output of each building block in a detector without disturbing the system.



(a) The first step is to extend a building block $BB(k)$ with input/output signal splitters that have scatter matrices $\mathbf{S}_{in,k}$ / $\mathbf{S}_{out,k}$. The number of input and output ports depend on the connection matrix \mathbf{C} .



(b) Connecting the nodes: if $\mathbf{C}(k, l)$ is not zero, then there is a connection from block l to block k , with strength $\mathbf{C}(k, l)$.



$$\mathbf{C} = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0.5 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0.2 & 0 \end{bmatrix} \quad \mathbf{S}_{in,0} = \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix} \quad \mathbf{S}_{in,1} = \begin{bmatrix} 0 & 1 & 1 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}$$

(c) An example network generated with sources/detectors on each element. The inset shows the used connection matrix and two of the $\mathbf{S}_{in,k}$ matrices.

Figure 4.13: Principle for creating a nanophotonic reservoir using Caphe. The neurons are encapsulated in building blocks (a), which are then connected to other blocks (b), in order to form a reservoir (c).

Most topologies in a neural network do not have a symmetrical connection matrix. This means that, if $BB(j)$ is connected to $BB(k)$, the signal does not necessarily propagate from $BB(k)$ to $BB(j)$. The strength of both connections can be controlled with $\mathbf{C}(j, k)$ and $\mathbf{C}(k, j)$, respectively. In photonics, however, a waveguide is by definition reciprocal, and signals can propagate back from $BB(k)$ to $BB(j)$. Therefore, connections are bidirectional and we choose $S_{in,k}$ to be symmetrical (i.e. we also fill the first column with $[0, 1, \dots, 1]$). For this reason, it is possible to enable bi-directionality between the nodes. In this case, the connections between the two building blocks are reciprocal. Also, the first column of $\mathbf{S}_{in,k}$ is filled except for $\mathbf{S}_{in,k}(0,0)$ (such that the matrix becomes symmetrical), which makes the splitter behave reciprocal.

After creating these extended building blocks, we use the information of the connection matrix to link them to each other, as illustrated in Figure 4.13(b). An example circuit with four components is illustrated in Figure 4.13(c).

4.6.2.2 Flattening the network

During the creation process we relied heavily on hierarchical structures which our framework supports naturally. This makes the creation process very flexible, less error-prone, and more natural, because each hierarchical component can be directly addressed by its ports without having to know the internal port numbering. It also allows us to re-use building blocks that are often used, which makes it much easier to create complex circuits. Also, the visualization of a circuit is more natural when using hierarchy, because hierarchical building blocks often represent a certain functionality.

However, hierarchical structures are MC structures by design, even if they consist solely of ML nodes⁶. As such, even if they contain ML nodes, which can be removed during the ML elimination step, the overall process is less efficient. For reasons of speed, ideally, there should be one large sparse generalized connection matrix \mathbf{S} in a top-level node. Also, when there is no hierarchy, the order in which the ODE equations of the sub-nodes are evaluated does not matter, which means it can be very easily parallelized. The process of removing all hierarchy is called flattening.

When flattening a network, we replace each hierarchical node by its sub-nodes, and re-link these sub-nodes properly to the rest of the circuit, in order to maintain the same topology. In the flattened network, the original hierarchical nodes does not exist anymore, instead we have a list of elementary building blocks (which cannot be further subdivided in other nodes), which can be ML or MC. In this network, we can first eliminate all ML nodes in order to perform efficient simulations in the time-domain, as we explained in section 4.4.2.

⁶This is a software-based limitation of the current architecture and will be resolved in version 2.0.

4.7 Alternatives

Converting this framework into a software package is a big effort. Before starting this effort we have made a comparison between developing our own package and using a commercial package such as VPI Photonics [8] or OptiSpice [10]⁷. However, there are four big limitations when using these products: first, it is difficult to create components with a specific behavior, as one is limited to the available building blocks. As we investigate new components, we regularly change the Ordinary Differential Equation (ODE) of the components (for example to incorporate the influence of temperature or free carriers). Second, the GUI does not allow us to easily create large networks. It is very cumbersome to create a network of 100 neurons with a random connection topology, and change the topology for each run. Third, the software has to be interfaced to a reservoir toolbox (such as the Matlab Reservoir Toolbox and the Python OGER toolbox) in an automatic way, and the software should be invoked from within this framework. Fourth, when optimizing any reservoir, we need to perform a huge number of simulations (a typical sweep has over 1000 simulations), which means the software needs to be run in parallel on a simulation grid. Purchasing 100 licenses to allow this was out of the question.

Then there is the important choice of programming language. The different building blocks can be mapped naturally on different objects, hence we only considered object-oriented programming languages. We have chosen to use the C++ programming language in favor of Matlab/Python. One could argue that Matlab and Python both have very good numerical libraries, but the fact is, C++ also has very good numerical libraries such as Eigen3⁸ which are very easy to use, and are comparable in performance⁹. Two other reasons why we chose this language were speed and easy interfacing:

1. Speed: Matlab and Python are interpreted languages, which means each line of code poses an overhead in calculation. A good Matlab/Python programmer will try to combine -where possible- for loops into vector or matrix operations in order to considerably speed up the calculations. However, when we combine different optical components, it is no longer possible to expand all component-specific computations (which is the bottleneck in a time-domain computation) in a vector operation. Hence, we lose all advantages of using fast matrix and vector operations in those

⁷During the third year of this PhD, a third package from Lumerical, called Interconnect, was launched.

⁸<http://eigen.tuxfamily.org/>

⁹Without any other dependency, it already performs extremely well compared to for example the Intel MKL (Math Kernel Library). Eigen3 can interface to other libraries, such as Intel MKL, when they are available. Furthermore, Eigen3 can cope very efficiently with sparse matrices, and supports parallelization for the most common matrix operations.

languages.

2. Easy interfacing and flexibility: C++ is easily interfaced to other programming languages thanks to the Simplified Wrapper and Interface Generator (SWIG)¹⁰. We have used SWIG in order to naturally map all C++ classes (optical components and solvers) onto Python classes. The C++ core bundled with the Python wrapper gives us an extreme flexibility and allowed us to couple the software framework to the reservoir computing toolbox OGER.

It turned out that with some simple modifications, we were able to extend the software framework so that it could calculate the frequency response of arbitrary circuits. Given that it would be very useful not only for reservoir computing, but for the nanophotonics community in general, we have had several refactoring rounds to clean up the code and make it available to non-reservoir computing users as well.

4.8 Conclusion

In this chapter, we presented a framework that enables modeling of optical circuits both in the time and in the frequency domain. It is suited for calculating the steady state characteristics of very large networks, and for modeling highly nonlinear systems in the time domain after eliminating linear instantaneous components. By eliminating these components, we reduce the effective size of the network, and the time-domain simulation becomes faster. The tool is very general and the internal variables can be expressed naturally depending on the application domain, which makes it attractive for other dynamical systems such as financial systems and neural networks. Furthermore, the framework supports hierarchical structures, which makes the network creation process more convenient and intuitive.

Although in this dissertation we have used the software framework mainly for nanophotonic simulations, it is already used frequently for other applications in photonics, such as in optical filter design. The software framework Caphe is therefore a very promising tool, as it is very fast, flexible, and can be combined with other scientific tools which are readily available in Python.

References

- [1] Ardavan F. Oskooi, David Roundy, Mihai Ibanescu, Peter Bermel, J. D. Joannopoulos, and Steven G. Johnson. *MEEP: A flexible free-software*

¹⁰<http://www.swig.org/>

- package for electromagnetic simulations by the FDTD method*. Computer Physics Communications, 181:687–702, January 2010.
- [2] Emmanuel Lambert, Martin Fiers, Shawkat Nizamov, Martijn Tassaert, and Wim Bogaerts. *Python bindings for the open source electromagnetic simulator MEEP*. Computing in Science and Engineering, 2010.
- [3] P. Bienstman and R. Baets. *Optical Modelling of Photonic Crystals and VCSELs Using Eigenmode Expansion and Perfectly Matched Layers*. Opt. Quantum Electron., 33:327–341, 2001. CAMFR simulation software is freely available from <http://camfr.sourceforge.net/>.
- [4] <http://camfr.sourceforge.net>.
- [5] <http://www.photonid.com/products/fimmwave.htm>.
- [6] Govind Agrawal. *Nonlinear Fiber Optics, Third Edition (Optics and Photonics)*. 2007.
- [7] <http://www.aspicdesign.com/>.
- [8] http://www.vpi Photonics.com/optical_systems.php.
- [9] <http://www.photonid.com/products/picwave.htm>.
- [10] <http://www.rsoftdesign.com/products.php?sub=System+and+Network&itm=OptSim>.
- [11] P. Gunupudi, T. Smy, J. Klein, and Z.J. Jakubczyk. *Self-Consistent Simulation of Opto-Electronic Circuits Using a Modified Nodal Analysis Formulation*. Advanced Packaging, IEEE Transactions on, 33(4):979–993, nov. 2010.
- [12] Tom Smy, Pavan Gunupudi, Steve Mcgarry, and Winnie N Ye. *Circuit-level transient simulation of configurable ring resonators using physical models*. America, 28(6):1534–1543, 2011.
- [13] <http://www.caphesim.com>.
- [14] Kristof Vandoorne, Wouter Dierckx, Benjamin Schrauwen, David Verstraeten, Roel Baets, Peter Bienstman, and Jan Van Campenhout. *Toward optical signal processing using Photonic Reservoir Computing*. Optics Express, 16(15):11182–11192, JUL 21 2008.
- [15] Piero Triverio, Stefano Grivet-Talocia, Michel S Nakhla, Flavio G Canavero, and Ramachandra Achar. *Stability, causality, and passivity in electrical interconnect models*. Advanced Packaging, IEEE Transactions on, 30(4):795–808, 2007.

- [16] How-Siang Yap. *Designing to digital wireless specifications using circuit envelope simulation*. In Microwave Conference Proceedings, 1997. APMC'97, 1997 Asia-Pacific, volume 1, pages 173–176. IEEE, 1997.
- [17] Shmuel Ben-Yaakov, Stanislav Glozman, and Raul Rabinovici. *Envelope simulation by SPICE-compatible models of linear electric circuits driven by modulated signals*. Industry Applications, IEEE Transactions on, 37(2):527–533, 2001.
- [18] Nuno B Carvalho, José C Pedro, Wonhoon Jang, and Michael B Steer. *Non-linear RF circuits and systems simulation when driven by several modulated signals*. Microwave Theory and Techniques, IEEE Transactions on, 54(2):572–579, 2006.
- [19] Thomas James Johnson. *Silicon microdisk resonators for nonlinear optics and dynamics*. PhD thesis, California Institute of Technology, 2009.
- [20] William H. Press, Saul A. Teukolsky, William T. Vetterling, and Brian P. Flannery. *Numerical Recipes: The Art of Scientific Computing*. 2007.
- [21] Timothy A. Davis and Ekanathan Palamadai Natarajan. *Algorithm 8xx: KLU, a direct sparse solver for circuit simulation problems*.
- [22] Timothy A. Davis. *Direct methods for sparse linear systems*. 2006.
- [23] Ken Stanley. *KLU: a Clark Kent sparse LU factorization algorithm for circuit matrices*. 2004 SIAM Conference on Parallel Processing for Scientific Computing (PP04), 2004.
- [24] Kristof Vandoorne, Joni Dambre, David Verstraeten, Benjamin Schrauwen, and Peter Bienstman. *Parallel reservoir computing using optical amplifiers*. IEEE Transactions On Neural Networks, 22(9):1469–1481, 2011.
- [25] N. Hansen. *The CMA evolution strategy: a comparing review*. In J.A. Lozano, P. Larranaga, I. Inza, and E. Bengoetxea, editors, Towards a new evolutionary computation. Advances on estimation of distribution algorithms, pages 75–102. Springer, 2006.
- [26] S.K. Selvaraja, W. Bogaerts, P. Dumon, D. Van Thourhout, and R. Baets. *Sub-nanometer Linewidth Uniformity in Silicon Nanophotonic Waveguide Devices Using CMOS Fabrication Technology*. Selected Topics in Quantum Electronics, IEEE Journal of, 16(1):316–324, jan.-feb. 2010.
- [27] F. De Leonardis, A. V. Tsarev, and V. M. N. Passaro. *Optical properties of new wide heterogeneous waveguides with thermo optical shifters*. Optics Express, 16(26):21333–21338, 2008.

- [28] Daoxin Dai, Liu Yang, and Sailing He. *Ultrasmall Thermally Tunable Microring Resonator With a Submicrometer Heater on Si Nanowires*. Light-wave Technology, Journal of, 26(6):704–709, march15, 2008.
- [29] Ciyuan Qiu, Jie Shu, Zheng Li, Xuezhi Zhang, and Qianfan Xu. *Wavelength tracking with thermally controlled silicon resonators*. Optics Express, pages 5143–8, 2011.
- [30] G Priem, P Dumon, W Bogaerts, D Van Thourhout, G Morthier, and R Baets. *Optical bistability and pulsating behaviour in Silicon-On-Insulator ring resonator structures*. Optics express, 13(23):9623–8, November 2005.
- [31] Thomas J Johnson, Matthew Borselli, and Oskar Painter. *Self-induced optical modulation of the transmission through a high-Q silicon microdisk resonator*. Optics express, 14(2):817–31, January 2006.
- [32] T. Van Vaerenbergh, M. Fiers, P. Mechet, T. Spuesens, R. Kumar, G. Morthier, B. Schrauwen, J. Dambre, and P. Bienstman. *Cascadable Excitability in microrings*. Optics Express, 20(18):20292–20308, 2012.
- [33] T. Van Vaerenbergh, M. Fiers, J. Dambre, and P. Bienstman. *Simplified description of self-pulsation and excitability by thermal and free-carrier effects in semiconductor microcavities*. Physical Review A, 86(6):063808, 2012.
- [34] H. Gruenberg. *Some Optimum Properties of n Ports*. Circuit Theory, IRE Transactions on, 8(3):329 – 334, sep 1961.
- [35] S. Ghosh, S. Keyvaninia, Y. Shoji, W. VanRoy, T. Mizumoto, G. Roelkens, and R. Baets. *Compact Mach-Zehnder Interferometer Ce: YIG/SOI Optical Isolators*. 2012.
- [36] Wim Bogaerts, Pieter Dumon, Dries Van Thourhout, and Roel Baets. *Low-loss, low-cross-talk crossings for silicon-on-insulator nanophotonic waveguides*. Opt. Lett., 32(19):2801–2803, Oct 2007.

5

Isolated spoken digit recognition using photonic crystal cavities

“One. ” Answer: two? (A reservoir)

In this chapter we will train a nanophotonic reservoir to recognize spoken isolated digits. This task was investigated previously in the doctoral thesis of K. Vandoorne [1], using a nanophotonic reservoir of Semiconductor Optical Amplifiers (SOAs). This was — to the best of our knowledge — the first time a nanophotonic reservoir computing system was proposed. In this chapter, we compare the previous results based on an SOA reservoir with our novel architecture based on Photonic Crystal Cavities (PhCC).

We will identify the relevant parameters and performance trends as a function of these parameters such as the phases between the resonators, the attenuation, the cavity lifetime, the delay between the resonators, the detuning and the input power. We also theoretically evaluate the influence of fabrication errors, which are currently the bottleneck for certain nanophotonic application domains.

This chapter is structured as follows: in section 5.1 we give a brief overview of the isolated digit recognition task. Then, in section 5.2 we summarize the most important results from K. Vandoorne’s dissertation. Because the behav-

ior of the SOA and the PhCC is very different, we compare the steady-state and time-domain characteristics of an SOA to those of a PhCC in section 5.3. In section 5.5 we then present experiments on the speech task using our novel architecture based on PhCC. We see a similar trend in performance as a function of interconnection delay (i.e., an optimal delay of approximately half the word duration), as we saw with classical reservoirs based on hyperbolic tangent neurons and the SOA reservoir. We will also show that the different topologies, introduced in 4.6.1, have a small influence. Furthermore, the type of cavity (inline versus side-coupled) has a big influence on the performance. Finally, there is a nonlinear region where the cavities are strongly coupled, which degrades the performance. We conclude in section 5.6.

5.1 Isolated digit recognition: task description

Speech recognition is a relatively complex task, which makes it a relevant benchmark for determining the overall performance of a reservoir. This type of task has been discussed in literature many times [2–5]. Also, reservoir computing has been used successfully on this task [6].

One benchmark task that is often used is the classification of isolated digits. This task was used in the PhD of Kristof Vandoorne [1], and by D. Verstraeten in [7]. Here, the words 'zero' to 'nine' are spoken by 5 female speakers. Each word is repeated 10 times. The samples are taken from the TI46 speech corpus [8]. It has become possible, using reservoir computing, to achieve a Word Error Rate (WER) close to 0%. This means that it is not possible anymore to clearly identify the influence of different parameters. For this reason additional babble noise was added. The source of this babble is 100 people speaking in a canteen, and the added noise has a Signal to Noise Ratio (SNR) of 3 dB [9].

5.1.1 Pre-processing

Instead of feeding the raw data from the speech task to the reservoir, we will pre-process the data, much like the human ear does. In his PhD thesis David Verstraeten compared different pre-processing methods [10]. He discovered that the Lyon passive ear model [11, 12] matches very well with the recognition capabilities of classical reservoir methods, and therefore this model was used in K. Vandoorne's and our experiments as well. Basically, the response of the inner ear and its selectivity to certain frequencies is modeled with a series of notch filters, followed by Half-Wave Rectifiers (HWR) and Adaptive Gain Controllers (AGC). The outputs are positive signals indicating the firing rate of the neurons. As a last pre-processing step, the samples are decimated with a factor 128 to reduce the number of samples without much loss of information. The resulting

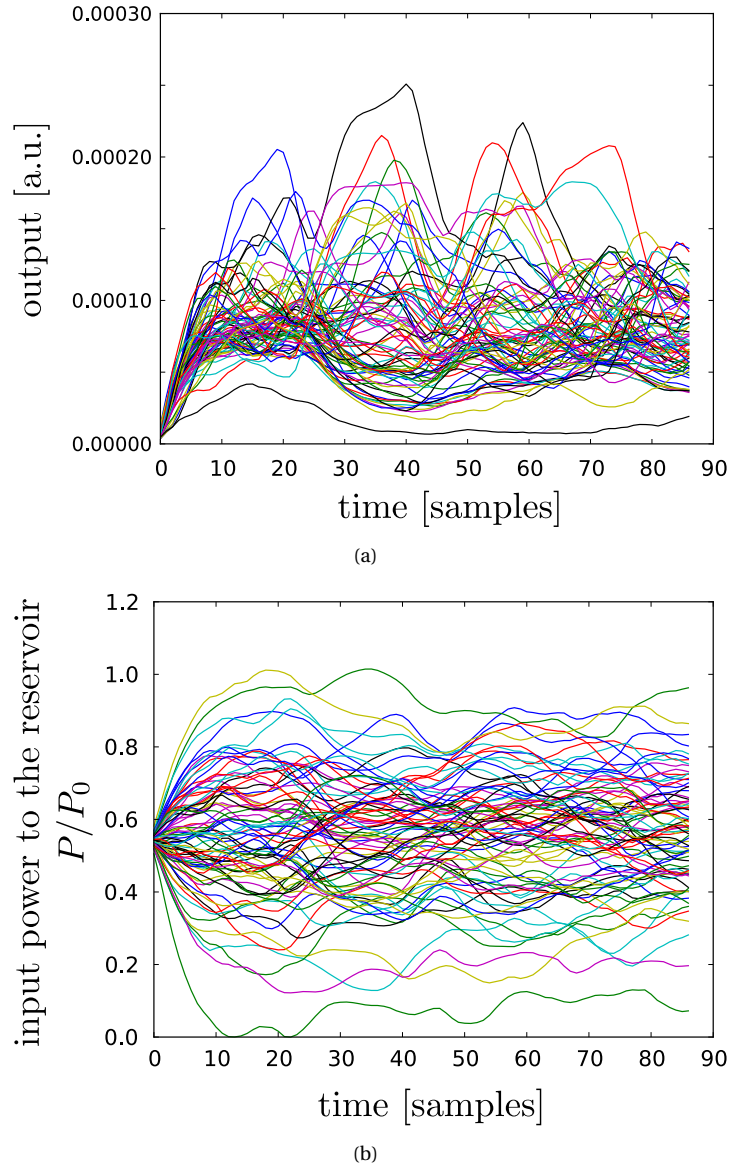


Figure 5.1: Example time traces for a spoken digit after pre-processing with the Lyon passive ear model. (a) shows the 77-channel output of the Lyon passive ear model for one spoken digit, and (b) shows the same data after multiplying with \mathbf{W}_{in} , and shifting all signals upwards, so they become positive for all timesteps. This is the actual input into the reservoir. In this example, we have normalized the output power such that the maximum power into a node is P_0 .

signal typically has a length of 40-80 samples.

In our case, one audio signal is transformed to 77 channels $\mathbf{u}_{channels}[k]$, where $k \in [0..K]$, for a spoken digit with K time samples. An example time trace is shown in Figure 5.1(a).

These 77 channels are multiplied with an input weight vector \mathbf{W}_{in} (see section 2.3.1) with dimensions $(81, 77)$, and weights randomly chosen from the set $\{-0.1, 0.1\}$. The final input to each of the 81 neurons is given by:

$$\mathbf{u}_{nodes}[k] = \mathbf{W}_{in} \cdot \mathbf{u}_{channels}[k], \quad (5.1)$$

This input is then shifted upwards such that all signals are positive, as we want to use this as input power for the photonic reservoir:

$$\mathbf{u}'_{nodes}[k] = \mathbf{u}_{nodes}[k] - \min_k(\mathbf{u}_{nodes}), \quad (5.2)$$

An example time trace is shown in Figure 5.1(b).

5.1.2 Winner-takes-all

We train ten distinct linear classifiers (see section 2.3.3), one for each digit. Each classifier should return +1 whenever its corresponding digit is spoken, and -1 otherwise. Figure 5.2 illustrates the winner-take-all approach: the winning digit corresponds to the classifier with the strongest positive response.

5.2 Previous work in nanophotonic reservoir computing using SOAs

In this section we briefly review the research that has been performed in the PhD thesis of Kristof Vandoorne [1]. The main focus of his dissertation was to demonstrate the use of SOAs in the context of reservoir computing.

However, it is not immediately obvious why such a network of SOAs makes a good reservoir. The steady-state input-output curve, although similar to the upper half of the hyperbolic tangent function which is classically used, lacks the symmetric lower branch because optical power cannot be negative. Also, the topology is restricted, because the chip is planar, and too many crossings should be avoided due to cross-talk and losses (this was discussed previously in 4.6). The increased complexity of combiners and splitters with a high fan-in resp. fan-out further increase the topology constraints.

On the other hand, advantages are that the SOA has richer internal dynamics as opposed to the static neurons which are used in software. As we alluded to in the introduction chapter, the fact that an optical signal has a phase and amplitude, will play an important role in the performance of the reservoir.

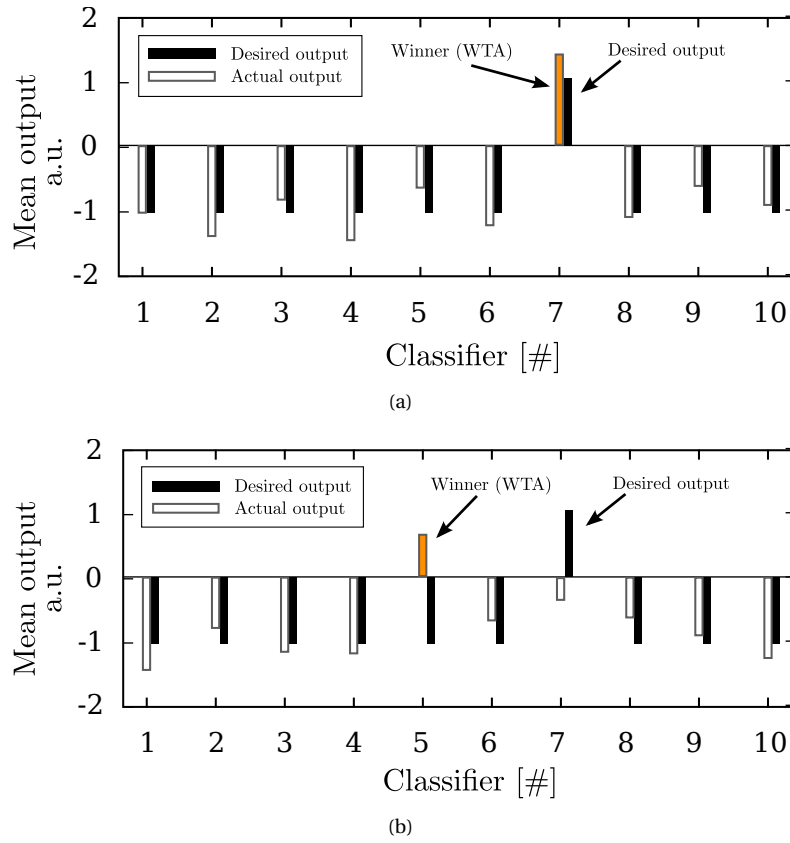


Figure 5.2: After taking the average over time of the output classifiers, we apply the winner-take-all principle. The winning sample is the one with the highest positive output. In (a), the highest output corresponds to spoken digit '7', which is correctly recognized. In (b), digit '5' is chosen as wrong answer.

We begin by introducing the SOA. We then explain which topology was used, and highlight the most important results that were achieved.

5.2.1 Semiconductor Optical Amplifiers

Semiconductor Optical Amplifiers (SOAs) can provide a high gain over short distances and can be electrically pumped. The operation of SOAs is similar to that of semiconductor lasers. Electrons from the valence band are excited to the conduction band, either through electrical or optical pumping. An incoming photon can interact with the excited electron, forcing the electron to release its energy and to return to the valence band. During the process, a new photon is emitted with exactly the same frequency, phase and direction. This process is called stimulated emission. More detail about the process can be found for example in [13]. SOAs are usually made from III-V compound semiconductors such as InP/InGaAsP and GaAs/AlGaAs.

5.2.2 Topology

Imagine a series of SOAs that are connected to each other on a single line. Information is always inserted from the left of each SOA, which means that information only flows from left to right. Because the SOA has no reflection, there is no communication from right to left. This is also true for a waterfall topology, as shown in Figure 4.11: SOA 2 does not influence SOA 1 if it is *after* SOA 2 (which means either right or down from SOA 1).

Because a unidirectional topology of non-reflecting components does not contain network memory¹, a swirl topology is used, as shown in Figure 4.7(a). In this topology, there are a large number of feedback cycles, which improve the connectivity between the individual SOAs. Because the SOAs amplify the input signals, an additional attenuation term is used in order to avoid lasing², or in other words, to avoid creating a loop somewhere in the circuit with a gain larger than one. Alternatively, the input current can be reduced to decrease the gain in the SOA.

¹For the PhCC, however, we can afford to use the waterfall topology, because there each cavity has a certain amount of reflection, which causes it to communicate to cavities on the left of, or above this cavity.

²With the current framework, laser operation, resulting from resonating cavities (where the cavity is modeled as an explicit building block), cannot be simulated correctly. This is mainly because for laser operation, we need multiple modes that operate at different frequencies, i.e. equation 4.8. This is not yet supported in the current version. This is different from the case where we see the laser as a single black box component with rate equations: this can be modeled in the current framework because the rate equations are ordinary differential equations.

5.2.3 Summary of previous results

The metric that is used to evaluate the performance of the reservoir is the Word Error Rate (WER), which defines the ratio between wrongly predicted digits versus the total number of available digits.

The main conclusion is that the SOA reservoir (i.e., with coherent simulations) outperforms the standard (incoherent) hyperbolic tangent reservoir (of the same size) for the isolated digit recognition task, for optimal parameters.

First, the influence of the delay between the neurons was investigated for both a classical ESN and the SOA network. As can be seen from Figure 5.3, the performance is better for an *optimal* delay that equals approximately half the duration of a spoken digit (in this case, this is a delay of 30 times the SOA delay).

Also, the effect of using amplitude and phase (called a coherent simulation) compared to using only the magnitude of the signal (called an incoherent simulation), is shown in Figure 5.3. The improved performance for a coherent simulation was explained as follows: the state space of a complex-valued reservoir contains twice as many variables as that of a real-valued reservoir. Although the number of observed variables remains unchanged (only the magnitudes are used for the readout layer), there is internal interaction between magnitudes and phases (through complex addition of signals). As a result, the additional richness of transformations in state space is also present in the magnitudes. Because the internal state-space is richer, the observed signals are more diverse and therefore more useful to approximate the desired output.

The attenuation between the connections is also relevant: the SOAs amplify the signal, and it has to be attenuated again in order to prevent lasing and/or chaotic behavior of the network. This behavior will most likely render the reservoir useless. This is equivalent to using a classical hyperbolic tangent discrete-time reservoir with a spectral radius that is very large (and larger than one).

5.2.4 Simulation framework

All research performed in the PhD of K. Vandoorne was done using the Matlab toolbox developed at the Department of Electronics and Information Systems (ELIS)³. Since then, a new framework was developed, called the OrGanic Environment for Reservoir computing (OGER) [14]. OGER, based on the free programming language Python, is an open-source machine learning toolbox based on the Modular toolkit for Data Processing (MDP) [15]. This framework allows one to easily run many simulations in parallel, along with a more modular approach which makes it easier to set up a simulation with fewer lines of code and better reproducibility.

³<http://reslab.elis.ugent.be/rctoolbox>

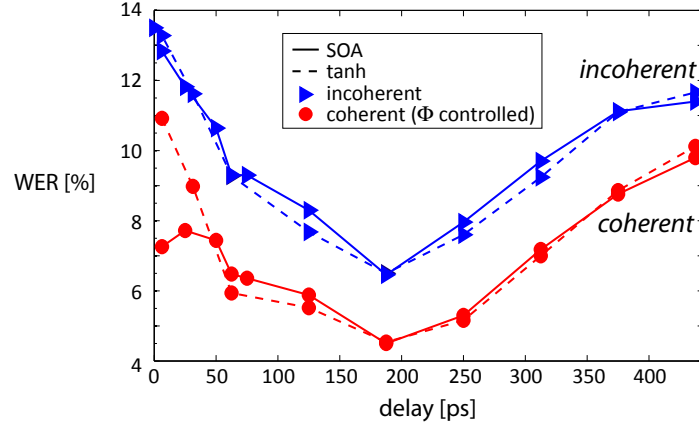


Figure 5.3: WER for the isolated digit recognition task, for a network of SOA and a classical hyperbolic tangent network. The SOA network, when simulated in a coherent regime (i.e., using complex-valued signals), performs better than a classical, real-valued (incoherent) hyperbolic tangent network. Clearly, there is an optimal value for the interconnection delay, which turns out to be approximately half of the word length (picture courtesy of K. Vandoorne).

5.3 Comparison between photonic crystal cavities and SOAs

In this section we compare the steady-state and dynamical behavior of the SOA and PhCC. The steady-state equation for the SOA is given by:

$$P_{out} = \exp(h)P_{in}, \quad (5.3)$$

in which h physically represents the integrated gain over the length of the SOA at steady-state, for a given input power [16]. Figure 5.4 shows the input-output relationship of the SOA in steady-state regime.

In section 3.2.1.2, the steady-state behavior of the PhCC was derived from the CMT equations and is given by:

$$\frac{P_{out}}{P_{in}} = \frac{1}{1 + (\Delta - P_{out}/P_0)^2}. \quad (5.4)$$

This equation is valid for an inline coupled cavity. For side-coupled cavities (where the light has a direct path from input to output, see Figure 3.4), the steady-state equation is given by [17]:

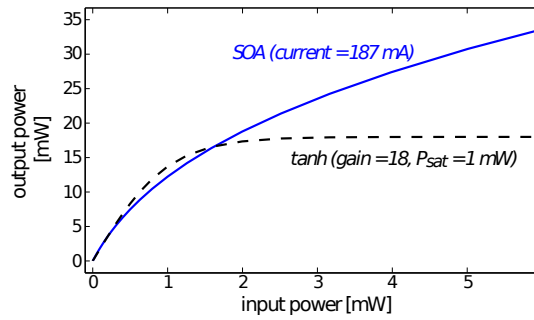


Figure 5.4: Input vs output power of an SOA. The input-output characteristic of this SOA resembles the input-output of a hyperbolic tangent function (picture courtesy of K. Vandoorne).

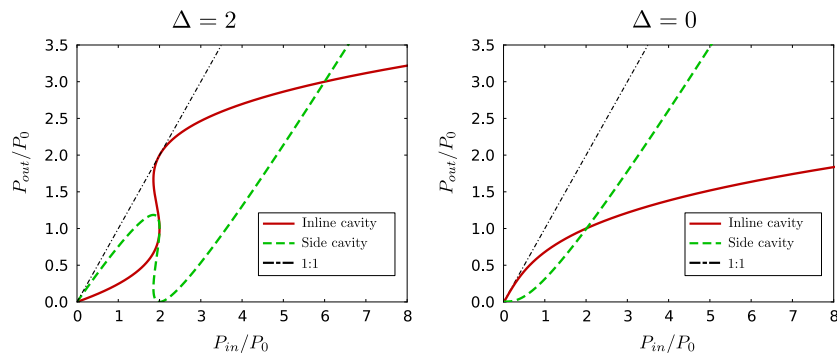


Figure 5.5: Input vs output power of the PhCC cavity. Two detunings are shown: a detuning $\Delta = 2$, for which bistability is observed, and a detuning $\Delta = 0$, which is on resonance. For both cases, an inline coupled (investigated in detail in chapter 3) and side coupled cavity are shown.

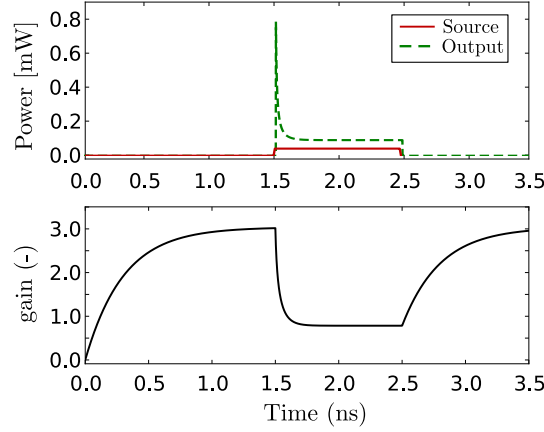


Figure 5.6: Step response of an SOA. After a warmup period of 1.5 ns, the source is turned on. The input signal is amplified by the SOA. For this it uses an amount of excited carriers, which decreases the SOA gain. When the source is switched off, the gain recovers to its steady-state value, which depends on the amount of current that is injected in the device.

$$\frac{P_{out}}{P_{in}} = \frac{(\Delta - (P_{in} - P_{out})/P_0)^2}{1 + (\Delta - (P_{in} - P_{out})/P_0)^2} \quad (5.5)$$

Figure 5.5 shows the steady-state behavior of both PhCCs. The bistable non-linearity in the PhCC for large positive detunings is more complex than the saturating nonlinear behavior of the SOA. Also, note the slope $P_{out}/P_{in} = h_{ss}$, at the origin, of the different components:

- SOA: $h_{ss} = \exp(g_0 L)$, where g_0 is defined as the small-signal gain of the SOA, which is directly proportional to $\frac{I}{I_0} - 1$, where I_0 is the current for reaching transparency. Furthermore, L is the length of the device. For an input current of 186 mW, $h_{ss} = 20.85$.
- Side cavity: $h_{ss} = \frac{\Delta^2}{1 + \Delta^2}$. For the simulations performed in this chapter, we have used detuning ranges from -2 to 2. For a detuning of $\Delta = \pm 2$ this corresponds to 0.8.
- Inline cavity: $h_{ss} = \frac{1}{1 + \Delta^2}$. For a detuning of $\Delta = \pm 2$ this corresponds to 0.2.

Also in the time-domain there are fundamental differences between an SOA and a PhCC. The step response of the SOA device is shown in Figure 5.6. Here we first have a warmup period where, by injecting current, the carrier density N increases until a steady-state value is reached. Afterwards, when a pulse arrives at

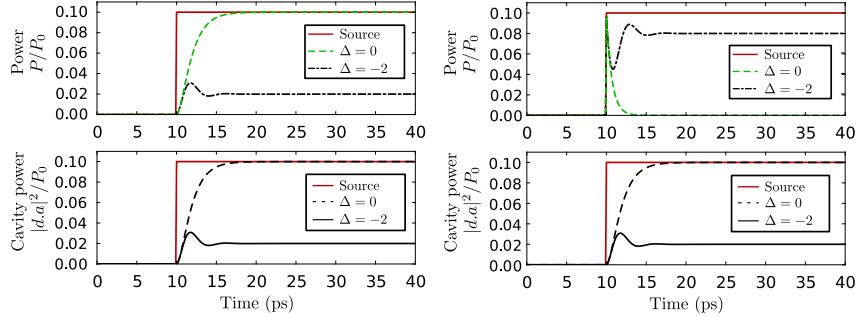


Figure 5.7: Step response of a photonic crystal cavity at low input powers ($P_{in} = 0.1P_0$). When excited at resonance ($\Delta = 0$), the inline coupled cavity (left) reaches a steady-state value close to unit transmission (it equals unit transmission in the limit $P_{in} = 0$, or when the Kerr nonlinearity is ignored). The side coupled cavity (right) has a steady-state value close to zero at resonance (and again, equals zero in absence of the Kerr nonlinearity).

the SOA, the light is amplified, and the carrier density decreases until it reaches a new steady-state value. This causes a large output signal immediately after the pulse was launched, which then decreases to a smaller steady-state value.

The photonic crystal cavity, in contrast, is a passive component. In Figure 5.7 we show the step response of an inline photonic crystal cavity. The resonator accumulates energy when the input is turned on. Depending on the detuning, different steady-state values are reached, and the observed output differs from the side-coupled and inline-coupled cavities.

5.4 The Jacobian of a system of photonic crystal cavities

As we saw in chapter 2, the spectral radius is defined as the largest eigenvalue of the system's Jacobian at its maximal gain state. This was defined in a discrete-time context. In the case of the hyperbolic tangent neuron $f(x)$, the function has a maximal gain of unity for $x = 0$. This means that in the small-signal limit, the Jacobian equals the weight matrix \mathbf{W}_{res} . Sometimes it is instructive to plot the spectrum of eigenvalues of this weight matrix. A spectral radius smaller than 1 means that all eigenvalues are within the unit circle. A purely linear system is stable if the spectral radius is strictly smaller than one⁴.

⁴Here we used the Bounded-Input, Bounded-Output (BIBO) stability criterion (the impulse response of the system is absolutely integrable).

When analyzing a continuous-time system, the system of equations that is solved is of the form:

$$\frac{d\mathbf{a}(t)}{dt} = \mathbf{f}(\mathbf{a}, \mathbf{s}_{in}, t) \quad (5.6)$$

Here, it is equally instructive to look at the eigenvalues of the Jacobian of this system, i.e., $\mathbf{J}(\mathbf{f})$. If the equations are linear, then the system is stable if all eigenvalues have a real value smaller than zero, i.e., the eigenvalues are in the left half of the complex plane.

In the next section, we derive the Jacobian for the PhCC devices. Since they are modeled using the (temporal) Coupled Mode Theory (CMT) equations (see section 3.2.1), the final Jacobian becomes very simple and elegant.

5.4.1 Deriving the Jacobian for a CMT system

Although we have defined the CMT equations in the case of a PhCC with one mode, the CMT equations can be generalized in order to describe an optical resonator with p ports and a finite set of modes (m modes) [18, 19]. In this derivation, we will assume that the system consists of a network of N resonators, without any additional components. As such, the \mathbf{s}_{in} , \mathbf{s}_{ext} and \mathbf{s}_{out} vectors have size N . In the actual simulations, we add sources to excite the system, which means that the \mathbf{s} vectors become larger, and the derivation more complex. When the system's Jacobian is calculated from within our software framework (see chapter 4), these sources can be eliminated and the Jacobian can be calculated numerically. The CMT equations for resonator i (in a network with N resonators) are given by (for simplicity assume each resonator has the same number of ports and modes):

$$\frac{d\mathbf{a}_i(t)}{dt} = \mathbf{M}_i \mathbf{a}_i(t) + \mathbf{K}_i^T \mathbf{s}_{i,in} + \mathbf{N}_i(\mathbf{a}, t, \dots) \quad (5.7)$$

Where \mathbf{M}_i ($m \times m$) describes the dynamics of the modes and the coupling between the modes. For example, for the PhCC with a single mode, M_i is a scalar value: $M_i = j(\omega - \omega_r) - 1/\tau$. \mathbf{K}_i ($p \times m$) describes the coupling from the inputs to the states. The function \mathbf{N}_i describes the nonlinear contribution (for example the Kerr nonlinearity, temperature effects, free carriers and so on).

The output of the resonator is given by:

$$\mathbf{s}_{i,out}(t) = \mathbf{S}_i \mathbf{s}_{i,in}(t) + \mathbf{D}_i \mathbf{a}(t), \quad (5.8)$$

where \mathbf{D}_i ($p \times m$) describes the coupling from the states to the output.

The state equation of the full system is given by

$$\frac{d\mathbf{a}(t)}{dt} = \mathbf{M}\mathbf{a}(t) + \mathbf{K}^T \mathbf{s}_{in} + \mathbf{N}(\mathbf{a}, t, \dots) \quad (5.9)$$

Here, the \mathbf{M} matrix is a block diagonal matrix, consisting of the individual \mathbf{M}_i , $i \in 0..N-1$. The \mathbf{K} and \mathbf{D} matrices are constructed in a similar way. \mathbf{s}_{in} is a concatenation of the individual $\mathbf{s}_{i,in}$, and similar for \mathbf{s}_{out} and the modes \mathbf{a} . Here, \mathbf{s}_{in} depends on the other resonators and on the topology of the circuit. We have derived the expression for the inputs $\mathbf{s}_{in}(t)$ in section 4.3.1. It is given by:

$$\mathbf{s}_{in}(t) = \mathbf{C}_{gen}\mathbf{s}_{ext}(t), \quad (5.10)$$

where \mathbf{C}_{gen} is the generalized connection matrix as derived in section 4.3.1. So by multiplying the generalized source term $\mathbf{s}_{ext}(t)$ we know the input at each of the resonators. In this case, the generalized source term is given by:

$$\mathbf{s}_{ext}(t) = \mathbf{D} \cdot \mathbf{a}(t) \quad (5.11)$$

Combining these expressions yields the ODE equation for the CMT system:

$$\frac{d\mathbf{a}(t)}{dt} = (\mathbf{M} + \mathbf{K}^T \cdot \mathbf{C}_{gen}\mathbf{D}) \mathbf{a} + \mathbf{N}(\mathbf{a}, t, \dots) \quad (5.12)$$

In the linear regime (i.e., when using low powers), the Jacobian \mathbf{J} is then given by:

$$\mathbf{J} = (\mathbf{M} + \mathbf{K}^T \cdot \mathbf{C}_{gen}\mathbf{D}) \quad (5.13)$$

5.4.2 Interpretation

A continuous-time system is stable if the eigenvalues of the Jacobian are in the left half of the complex plane at all times⁵ (In discrete-time systems, this holds if all eigenvalues are inside the unit circle). The real part of the rightmost eigenvalue of the Jacobian then gives information about how fast information leaks away in the reservoir. However, it only tells something about the stability if the system is linear. During this chapter, we sometimes visualize this spectrum of eigenvalues as an illustration. In these cases, the corresponding experiments were performed with very low powers, i.e., the reservoir is quasi linear.

In a nonlinear system, the Jacobian depends on the actual state of the reservoir. In order to draw conclusions about the stability (or chaoticity) in this case, one can use the Lyapunov exponent [10, 20]. It is a measure for the exponential deviation from a trajectory, which results from an infinitesimal disturbance of the system state.

⁵However, it does not mean that the system is necessarily unstable if some of the eigenvalues are sometimes to the right of the imaginary axis.

Name	Description	Default value
width	Width of the 2D mesh	9
height	Height of the 2D mesh	9
N	Number of cavities	81
BF	Bias fraction (fraction of nodes that receive bias)	0.0
ϕ_j	Phase reflection of the cavities	0.2π
P_{max}	Maximum neuron input power	$0.05P_0$
α	Attenuation between the connections	0 dB
λ	Signal wavelength	1551.83 nm
λ_r	Cavity resonance wavelength	$1550 \text{ nm} \pm 0.2 \text{ nm}$
τ	Cavity lifetime	0.139 ps / 1.25 ps
Δ	Detuning of the cavity ($= \tau(\omega - \omega_r)$)	2 ± 0.217
τ_d	Delay between the resonators	[0-6] ps
Δt	Speech sampling period	0.1 ps

Table 5.1: Default values used in the photonic crystal cavity (PhCC) reservoir for the isolated digit recognition task.

5.5 Simulation results

In this section we describe the experiments that were performed for the isolated digit recognition task using a Photonic Crystal Cavity nanophotonic reservoir. We will discuss the influence of delay and the cavity lifetime, the effect of randomized phases and fixed phases, the influence of the topology and cavity type, and the influence of the input scaling⁶ and detuning, as this influences the non-linear contribution due to the Kerr nonlinearity.

5.5.1 Default parameters

The default network has 81 resonators, in a 9x9 regular mesh, similar to the SOA reservoir which we want to compare to. There is no input bias, and we choose a very low input scaling IS (which corresponds to the linear regime). The detuning of the cavities is $\Delta=2$, a value which, for higher powers, gives rise to non-linear dynamics. Most of the experiments however are conducted in a regime with low powers, i.e., $P_{max} = 0.05P_0$. For the chosen variation of λ_r , the detuning can change by 0.217 (this variation was calculated for a fixed τ). The default parameters for all simulations performed in this chapter are summarized in Table 5.1.

⁶For the definition, see section 2.3.1, paragraph about constructing the weight matrices.

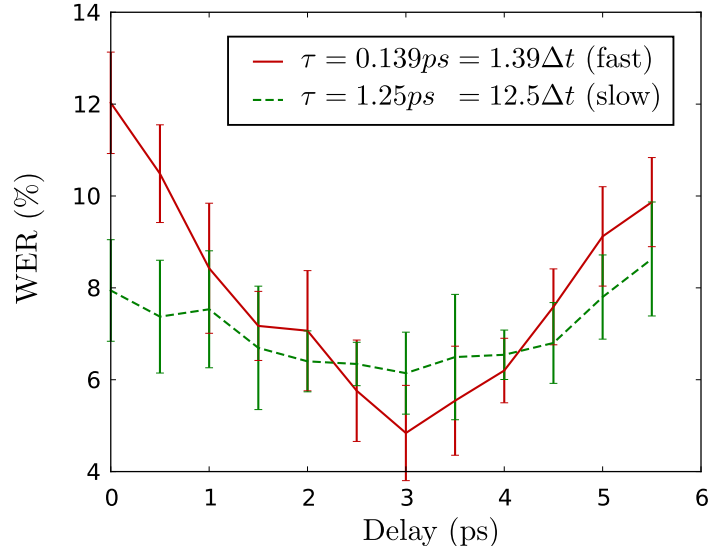


Figure 5.8: WER as a function of the interconnection delay τ_d , for $\tau = 1.25 ps = 12.5\Delta t$ (slow) and $\tau = 0.139 ps = 1.39\Delta t$ (fast), and a network with randomized phases. For the cavities with a small lifetime (small τ), the WER shows a clear optimum when the delay is approximately half the duration of a spoken digit. This is in correspondence with previous results using a reservoir of SOAs, which showed the same type of optimum. The optimal WER of 5 percent is comparable to the WER of 4.5 percent that is found for the SOA network. The simulations were performed for a waterfall topology.

The sampling period of the speech signal was set to $\Delta t = 0.1$ ps. The reservoir can be simulated internally with a smaller timestep, to ensure the simulations are accurate and stable (typically this simulation step is a fraction of the cavity lifetime, e.g., $\tau/10$). Also, in this chapter we have mainly used the Runge-Kutta 4 integration method, which is more accurate than the Euler method (see section 4.3.2).

Each simulation was performed ten times, each time with a different random initialized input matrix \mathbf{W}_{in} . The plots show the average over the 10 runs, while the error bars show the sampled standard deviation.

5.5.2 Influence of the cavity lifetime

An important finding of the research performed by K. Vandoorne is that the interconnection delay, i.e., the delay between the individual neurons, plays a crucial role for the performance of the isolated digit recognition task. This was

summarized in Figure 5.3. Figure 5.8 shows that this is also the case for the in-line photonic crystal cavity if the cavity lifetime τ is chosen small compared to the speech signal sampling period Δt .

We can explain this as follows. The green dashed curve in Figure 5.8 is for the case of a long cavity lifetime compared to the sampling period: $\tau = 1.25 ps = 12.5\Delta t$. This means that the energy decays to $1/e$ of its original value in about approximately 12 time steps. In this case, there seems to be sufficient memory in the nodes themselves, and the influence of interconnection delay (i.e. interconnection memory) is less pronounced. The second experiment used cavities with a small cavity lifetime ($\tau = 1.39\Delta t$)⁷. In this case, the neurons have almost no memory, and all the memory in the system has to come from the interconnections. Now there is a clear optimal delay around 3 ps, i.e. 30 samples. This corresponds to approximately half the duration of a typical speech sample, which range between 45 and 90 samples). This optimal delay is in good agreement with previous results using an SOA reservoir, where the individual SOAs do not possess leak rate (they were used in a pseudo-static regime), but rather amplify the input in a nonlinear way. There, also an optimum around 30 samples was found. Also, the trend for longer τ more closely resembles the results for leaky hyperbolic tangent networks [1]. The WER of 4.8% for a delay of about 3 ps is the best performance that is reached in this chapter.

5.5.3 Fixed phase versus random phase

In Figure 5.8 which we discussed earlier, the phases of the interconnections between the resonators were uniformly drawn from the range $[0, 2\pi]$ (which we will call *randomized phases* from now on). Here, we investigate what happens when the phase reflections, ϕ (as defined in section 3.2.1), are identical for all resonators (which we will call *fixed phases*). In Figure 5.9, the thin lines represent different fixed phases ϕ and the thick line represents the previous experiment with random phases.

As can be seen from the figure for fixed phases, the performance of the reservoir is greatly influenced by the chosen phase between the resonators. This effect is more pronounced when the cavities are slower (i.e., when they have a larger cavity lifetime τ), as can be seen in Figure 5.9. This can be explained as follows: there is more interaction between cavities when they have a large cavity lifetime. Depending on the phase difference between the cavities, the dynamics can be different, leading to different behavior (Figure 5.9, bottom). On the other hand, for fast cavities, there is less interaction between the cavities, hence it is less susceptible to the actual topology (Figure 5.9, top).

⁷In this case, the internal integration timestep in Caphe was chosen smaller to ensure stability of the simulation.

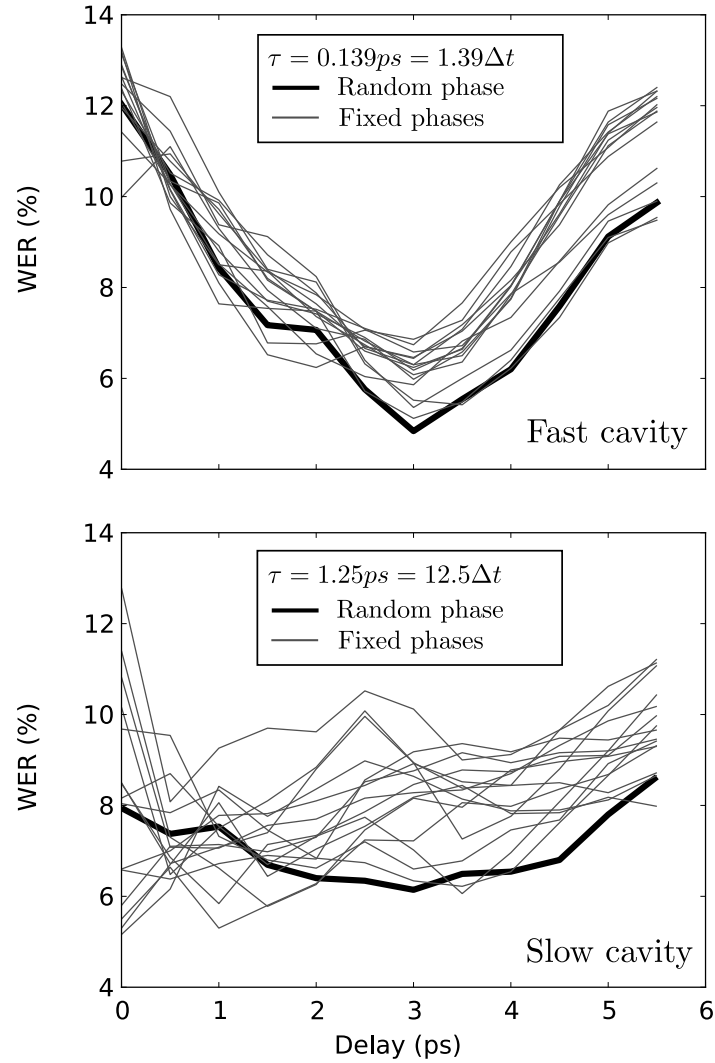


Figure 5.9: WER for fixed phases, again in the case for $\tau = 0.139 \text{ ps} = 1.39 \Delta t$ (fast, top) and $\tau = 1.25 \text{ ps} = 12.5 \Delta t$ (slow, bottom). The thick lines are the result for random phases (the same as in Figure 5.8), the thin lines are the results for different fixed phases (i.e. $0, 0.1\pi, 0.2\pi \dots$). As can be seen from the figure, the influence of the phase on slow resonators is larger, and covers a wider band. For some phases, the performance for small delays is comparable to the optimal performance for a delay of approximately 3 ps.

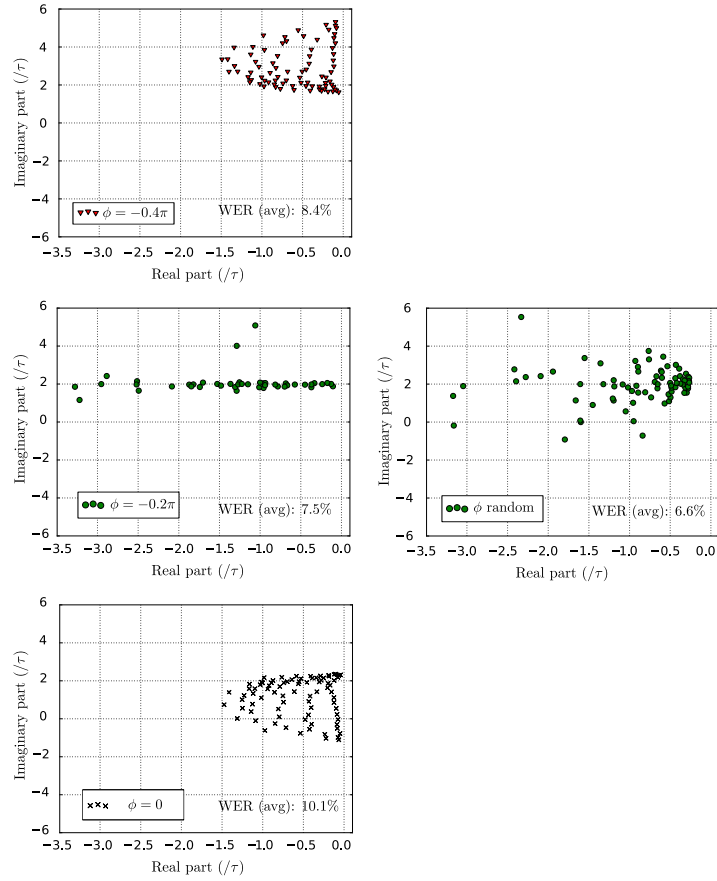


Figure 5.10: The eigenvalue spectrum of the Jacobian for a network of in-line coupled resonators, for fixed phases (left), and random phases (right). For fixed phases, depending on the actual phase reflection of the cavities, the spectrum can be completely different. The reservoir performance depends heavily on the actual phase that is used. The WER that are mentioned correspond to the slow cavities, for $\tau_d = 2.5ps$. The spectrum for the random phases on the right is shown as an illustration.

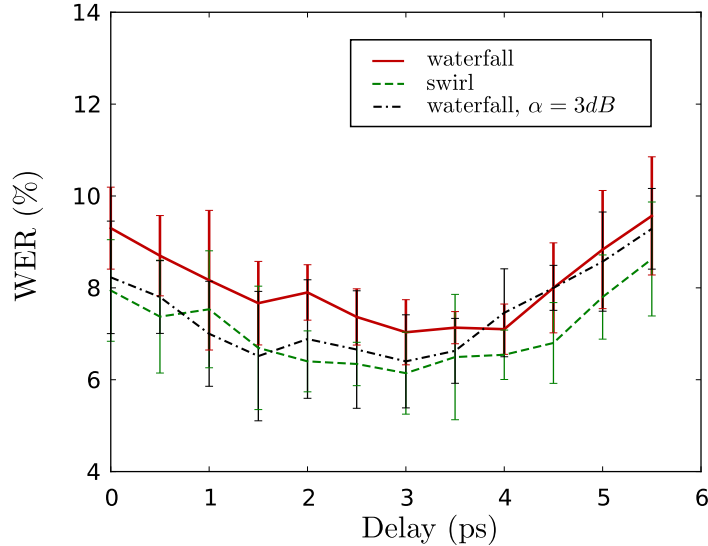


Figure 5.11: WER as a function of the interconnection delay τ_d , for $\tau = 1.25ps = 12.5\Delta t$. For the waterfall topology, the rightmost eigenvalues of the Jacobian is closer to the origin than in the case of the same topology with an attenuation of 3 dB, or the swirl topology. As a result, because the eigenvalues are very close to the origin, the reservoir is less responsive to the inputs and the performance is less good for the waterfall topology without attenuation.

For random phases, there is an additional richness in the reservoir because the interactions are different between different cavities. For that reason, the overall performance is better.

As an illustration we show the eigenvalue spectrum for three different values of ϕ in Figure 5.10. It is clear that the behavior can be very different depending on the actual phases that were used.

5.5.4 Influence of the topology

Figure 5.11 shows that the performance is better for the swirl topology for all values of the delay. This mainly has to do with the fact that the performance also depends on the rightmost eigenvalue of the Jacobian, which greatly determines how strong the system responds to the input. This is similar to the well-known assumption that a reservoir should be at the “edge of stability” in order to perform well. Depending on the task, the actual dynamic region where the reservoir performs best can be different. It is important to note that the initial connection weights for the swirl topology are chosen different (i.e., smaller

in amplitude) than those of the waterfall topology. Because of this, the initial rightmost value of the Jacobian for the swirl topology is more to the left than for the waterfall topology for the same value of the attenuation α . If we look at the eigenvalue spectrum in Figure 5.12, one can see that the smallest eigenvalue is close to the imaginary axis for the waterfall topology. In this case, the waterfall topology is less responsive to the inputs and the word error rate increases. Using some attenuation (α) (which initially shifts the eigenvalues to the left), makes the reservoir more responsive. This is also shown in Figure 5.11 for $\alpha=3$ dB.

5.5.5 Influence of the cavity type

In this section we investigate whether the cavity type (i.e., inline versus side-coupled cavities) has a large influence. The scatter matrix of the two types of cavities is different, hence it will have a large influence on the generalized connection matrix of the system $\mathbf{C}_{exttoin}$. Instead of again sweeping the delay, we now sweep the attenuation α , because it also influences $\mathbf{C}_{exttoin}$. We also conduct our experiments both for the swirl and waterfall topology. Figure 5.13 summarizes the results. In this experiment, the detuning $\Delta = 0$, and there is no interconnection delay τ_d . The phase was chosen fixed and $\tau=0.695$ ps (which is a value in between the slow and fast cavity which we studied previously).

There is a clear difference in performance for the side-coupled cavity (which has a bad WER overall) and the inline cavity. This can be explained as follows: for the given detuning $\Delta = 0$, the side-coupled cavity has a steady-state transmission of zero. This means the side-coupled cavity tries to forget the information that it was given as input. Although not shown on the figure, this effect becomes worse when the cavity lifetime is smaller, because in this case the steady-state value is reached sooner. In the limit where the lifetime is much smaller than the signal sampling period (and becomes zero), the output of the neurons is zero, hence the reservoir is unable to process information.

In [21] and in previous work by K. Vandoorne, it was shown that the actual topology is not so relevant for the performance of a reservoir. Figure 5.13 confirms that the difference between a waterfall and swirl topology, for this specific task, is largely irrelevant, especially for higher values of the attenuation. However, it must be noted that the rightmost eigenvalue of the Jacobian is different for both topologies (in case of the swirl topology, it is slightly more negative because the splitters introduce more loss), and this causes the lower performance of the waterfall topology for 0 dB attenuation in the connections. This is verified by looking at the eigenvalue spectrum of the Jacobian (which we defined in section 5.4.1), in Figure 5.12, and this is the same effect as we discussed in section 5.5.4.

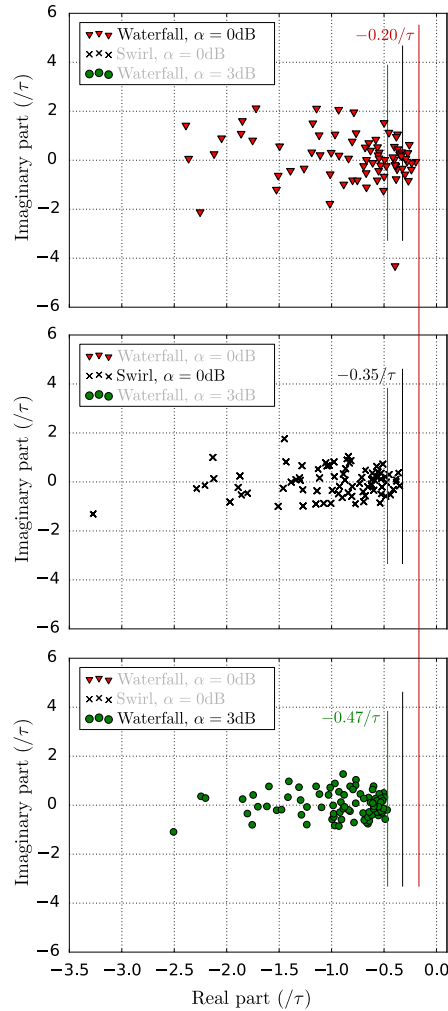


Figure 5.12: The eigenvalue spectrum of a system with random phases and $\Delta = 0$. Increasing the attenuation will shift the rightmost eigenvalue towards the left. When the attenuation is very large, all eigenvalues will end up at $(-1/\tau, 0)$. The waterfall with no attenuation has a rather high 'spectral radius'. For the isolated digit recognition task however, this value should not be too high. This explains why the waterfall topology without attenuation, as shown in Figure 5.8, has a lower performance.

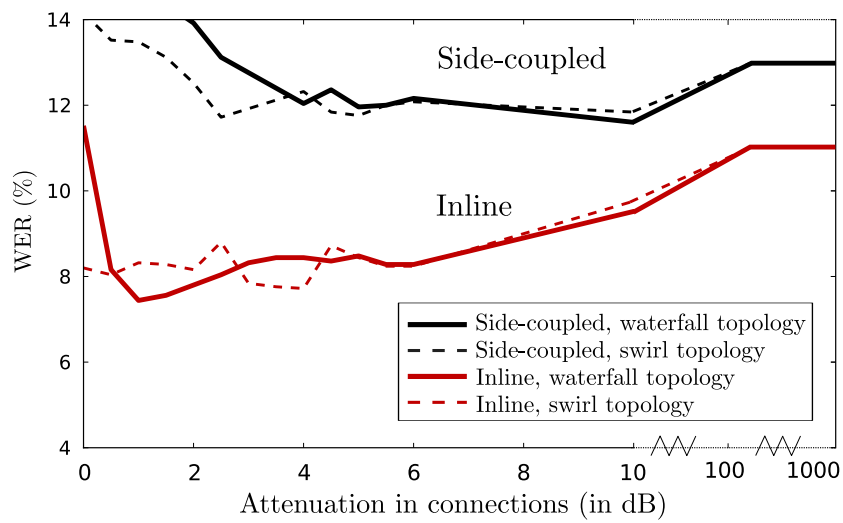


Figure 5.13: Word Error Rate (WER) as a function of the attenuation, for side-coupled and inline cavities, for different topologies. The side-coupled cavities perform less than the inline coupled cavities, because their transmission at steady-state is equal to zero (see Figure 5.7). As can be seen from the figure, the topology (swirl vs waterfall) does not influence the performance much, except for low values of the attenuation. The phase was chosen fixed, $\tau=0.694835$ ps, $\Delta=0$ and $\tau_d=0$ ps.

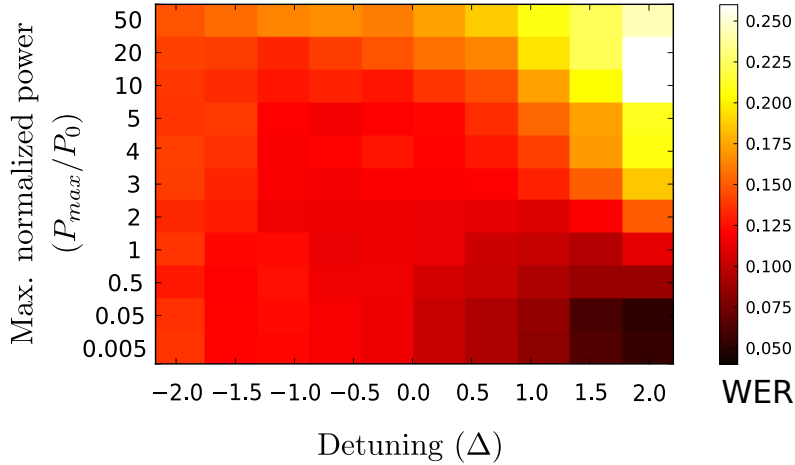


Figure 5.14: WER as a function of the the detuning and input power of the reservoir. For large detunings, the dynamics of the cavity are not so dependent on the power (this is because the resonance shape of the cavity resonance skews towards negative detuning as shown in Figure 5.15). For positive detunings ($\Delta > 1$), the nonlinearity in the system increases with higher powers, which decreases the performance of the reservoir. Note that the very good performance for low power and $\Delta = 2$ is highly dependent on the actual phase that was chosen between the cavities. In this case, it corresponds to the lowest curve of Figure 5.9.

It is interesting to note that, for both the inline cavity and side cavity, there is no loss mechanism in the cavity itself. In reality, because the photonic crystal cavity is a three-dimensional structure, light will scatter out-of-plane and there will always be some loss. This can be quantified by measuring the transmission loss. From the measurements performed in section 3.3.3, we have found a best-case transmission loss of 2.3 dB. This loss can be translated, to a good degree of accuracy, to the attenuation parameter which we discussed here. In other words, from Figure 5.13, the restrictions on the transmission losses are less stringent for the performance of the reservoir. This is in contrast to other application domains in nanophotonics, where typically one pursues cavities with very large Q-factors and low losses.

5.5.6 Influence of detuning and input power

Figure 5.14 shows the performance as a function of the detuning $\Delta = (\omega - \omega_r)\tau$ and the maximal power that each neuron receives. If neuron i receives an input signal with power $P_i(t)$, then $P_{max} = \max_{i,t}(P_i(t))$ is the highest power that

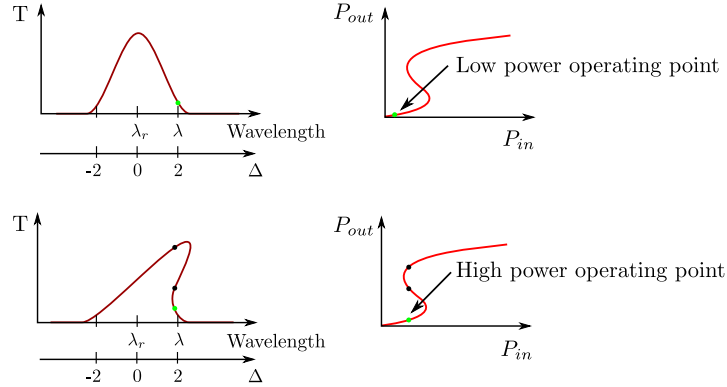


Figure 5.15: The transmission as a function of wavelength for different input powers. For higher input powers, bistability is observed.

ever arrives at a neuron. This is calculated over all neurons, over all samples of in the dataset. Then the power is normalized w.r.t. the characteristic nonlinear power P_0 (which defines the strength of the Kerr nonlinearity). This maximum power can be controlled by changing the input scaling of the reservoir (i.e., the factor with which we multiply the input weights \mathbf{W}_{in} of the reservoir).

These parameters are strongly connected, because nonlinearities occur at high powers for certain detunings. The detuning Δ determines the degree of off-resonance (a small value of the detuning means close to resonance). As we saw in chapter 3, there are regions where the system starts self-pulsating, and there are regions where the system becomes chaotic. This was investigated for a small system of two coupled cavities, and also for three coupled cavities in our paper [22]. The amount of nonlinearity in the reservoir increases for positive detunings with a large magnitude (i.e., $\Delta > 1$). This is because for high input powers, the Lorentzian shape of the resonance skews towards larger detunings, as shown in Figure 5.15. This means that at some point for high input powers a bistability occurs, which radically changes the type of nonlinearities in the system. In our experiments we observed that using positive detunings around $\Delta = 2$, can cause the system to behave in a much more nonlinear fashion (the system becomes chaotic or starts self-pulsing for fixed input). As the speech recognition task is mostly a linear task, the performance is worse for these detunings with high input powers, as was shown in Figure 5.14. In the next chapter we will investigate the total information processing capacity of this system, which will show that under these circumstances, the total processing capacity (which also includes the nonlinear part of the capacity) will decrease.

5.5.7 Influence of fabrication errors

In this last experiment we check the influence on the performance of random variations in the lifetime τ , and in the resonance frequency λ_r . The actual values of the parameters are perturbed by a value drawn from the normal distribution with unit standard deviation, i.e., τ is perturbed with $\tau_{rand}N(0, 1)$, and ω_r is perturbed by $\omega_{rand}N(0, 1)$.

For the silicon on insulator platform, a resonance variation of 0.5 nm is common for ring resonators, and we have measured a variation of 5 nm for the photonic crystal cavities (see section 3.3.3). Fortunately, as can be seen from Figure 5.16, the influence of the resonance frequency is small. However, this is not always the case. In the next chapter, we will see that the influence of these wavelength variations will be larger for the performance on a signal generation task.

The influence of the variability of the cavity lifetime τ is more detrimental, as can be seen in Figure 5.17. In these simulations, a lower cut-off value of 0.3 ps was used, i.e. the fastest cavity had a lifetime $\tau = 0.3$ ps. As can be seen, the maximum randomness that is allowed for still having a good performance is proportional to the cavity lifetime itself. In other words: for slower cavities, more randomness is allowed. To give an idea for the photonic crystal cavities which we measured in section 3.3: we found Q-factors of approximately 500 (which, for $\lambda = 1.55\mu m$ corresponds to $\tau=0.82ps$ ⁸) and 20000 ($\tau=32.9ps$), with an average of $\tau=10.76$ ps and a standard sampling deviation of 11.29 ps. This means we first need to decrease the variability on the quality factors of the resonators before being able to successfully train this type of reservoir.

5.6 Conclusions

In this chapter, we have investigated a novel reservoir architecture based on photonic crystal cavities by training the reservoir to recognize isolated digits (the isolated digit recognition task). The research in this chapter is based on the experience and previous work in [23], where the isolated digit recognition task was tested for a reservoir of semiconductor optical amplifiers.

We have performed a series of experiments to discover trends in the very large parameter space, and have found a performance that is similar to [23], which is state-of-the-art, and which is better than the performance of a classical hyperbolic tangent reservoir for the same task. We have found that the exact topology is of less importance (swirl vs waterfall), as was shown previously ([21], [1]). The interconnection delay is an important parameter, and leads to an optimal performance when it is half the duration of a spoken digit. This is similar to the results in [23]. For cavities with a large leak rate (i.e., slower), the

⁸According to $\tau = \frac{2Q}{\omega}$.

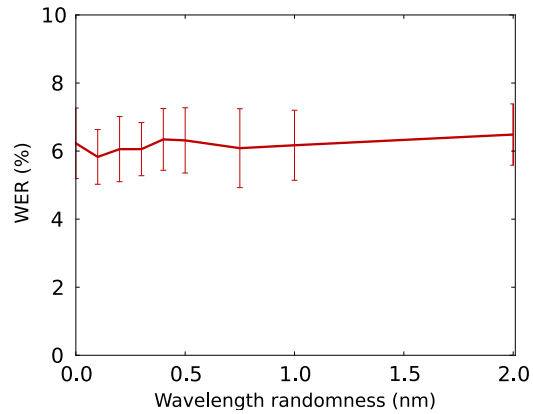


Figure 5.16: WER as a function of the variation ω_{rand} in the resonance frequency. The variations in ω do not have a significant effect. The interconnection delay $\tau_d=0$ ps.

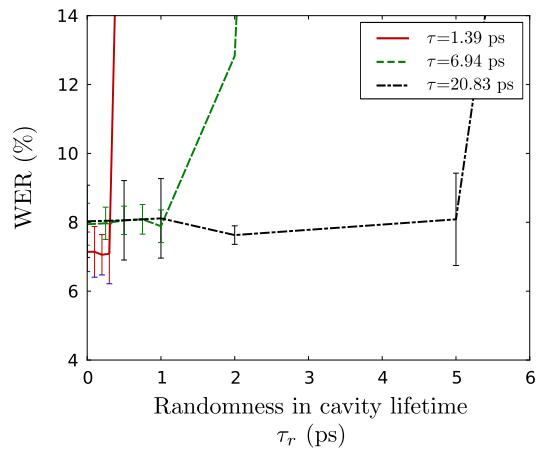


Figure 5.17: WER as a function of the variation τ_{rand} in the lifetime of the cavity, for different mean cavity lifetimes. When increasing the mean cavity lifetime, we can afford a larger randomness in the cavity lifetime. The interconnection delay $\tau_d=0$ ps.

performance as a function of this delay is less pronounced, and the reservoir is more sensitive to phase variations.

In our experiments, side-coupled cavities lead to worse performance than inline cavities. This mainly has to do with the fact that side-coupled cavities have a steady-state low-signal gain of zero. In other words, they try to annihilate the information.

Although this chapter used components with a Kerr type nonlinearity, the results can probably be extended to other nonlinearities which are common for the SOI platform, such as temperature effects and the plasma dispersion effect.

Finally, we have investigated the influence of random process variations on the performance of the reservoir. Because optical components built on the silicon photonics platform are usually very sensitive to small fabrication errors, it is often very difficult to design a device or component with good performance. Fortunately, a reservoir is very robust (compared to other nanophotonic applications) to variations in the topology, and, in our case, to the resonance frequencies and the leak rates of the cavities.

References

- [1] Kristof Vandoorne. *Photonic Reservoir Computing with a Network of Coupled Semiconductor Optical Amplifiers*. PhD thesis, Ghent University, 2011-2012.
- [2] A. J. Robinson, G. D. Cook, D. P. W. Ellis, E. Fosler-Lussier, S. J. Renals, and D. A. G. Williams. *Connectionist speech recognition of Broadcast News*. *Speech Communication*, 37(1-2):27–45, 2002.
- [3] A. J. Robinson. *An Application of Recurrent Nets to Phone Probability Estimation*. *IEEE Transactions on Neural Networks*, 5(2):298–305, 1994.
- [4] A. I. García-Moral, R. Solera-Ureña, C. Peláez-Moreno, and F. Díaz de María. *Data Balancing for Efficient Training of Hybrid ANN/HMM Automatic Speech Recognition Systems*. *IEEE Transactions on Audio, Speech & Language Processing*, 19(3):468–481, 2011.
- [5] A. I. García-Moral, R. Solera-Ureña, C. Peláez-Moreno, and F. Díaz de María. *Hybrid Models for Automatic Speech Recognition: A Comparison of Classical ANN and Kernel Based Methods*. In *Proceedings of Advances in Nonlinear Speech Processing (NOLISP)*, Lecture Notes in Computer Science, pages 152–160, 2007.

- [6] M. D. Skowronski and J. G. Harris. *Automatic speech recognition using a predictive echo state network classifier*. *Neural Networks*, 20(3):414–423, 2007.
- [7] D. Verstraeten, B. Schrauwen, and D. Stroobandt. *Isolated word recognition using a liquid state machine*. In *Proceedings of the 13th European Symposium on Artificial Neural Networks (ESANN)*, pages 435–440, 2005.
- [8] G. R. Doddington and T. B. Schalk. *Computers - Speech Recognition - Turning Theory to Practice*. *IEEE Spectrum*, 18(9):26–32, 1981.
- [9] A. Varga and H. J. M. Steeneken. *Assessment for automatic speech recognition: II. NOISEX-92: A database and an experiment to study the effect of additive noise on speech recognition systems*. *Speech Communication*, 12(3):247–251, 1993.
- [10] D. Verstraeten. *Reservoir Computing: Computation with Dynamical Systems*. Phd, Ghent University, 2009.
- [11] R. Lyon. *A computational model of filtering, detection, and compression in the cochlea*. In *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, volume 7, pages 1282–1285, Paris, France, May 1982.
- [12] M. Slaney. *Lyon's Cochlear Model*. Apple Computer Technical Report #13, 1988. <http://www.slaney.org/malcolm/pubs.html>.
- [13] *Fundamentals of Photonics*. John Wiley and Sons, Inc., New York, USA, 1991.
- [14] David Verstraeten, Benjamin Schrauwen, Sander Dieleman, Philemon Brakel, Pieter Buteneers, and Dejan Pecevski. *Oger: Modular Learning Architectures For Large-Scale Sequential Processing*. *Journal of Machine Learning Research* (submitted), 2011.
- [15] Tiziano Zito, Niko Wilbert, Laurenz Wiskott, and Pietro Berkes. *Modular toolkit for Data Processing (MDP): a Python data processing framework*. *Frontiers in Neuroinformatics*, 2(00008), 2009.
- [16] G.P. Agrawal and N.A. Olsson. *Self-Phase Modulation and Spectral Broadening of Optical Pulses in Semiconductor-Laser Amplifiers*. *IEEE Journal of Quantum Electronics*, 25:2297–2306.
- [17] Mehmet Fatih Yanik, Shanhui Fan, and Marin Soljacic. *High-contrast all-optical bistable switching in photonic crystal microcavities*. *Applied Physics Letters*, 83(14):2739–2741, oct 2003.

- [18] S. Fan, W. Suh, and JD Joannopoulos. *Temporal coupled-mode theory for the Fano resonance in optical resonators*. JOSA A, 20(3):569–572, 2003.
- [19] W. Suh, Z. Wang, and S. Fan. *Temporal coupled-mode theory and the presence of non-orthogonal modes in lossless multimode cavities*. Quantum Electronics, IEEE Journal of, 40(10):1511–1518, 2004.
- [20] Kathleen T Alligood, Tim D Sauer, and James A Yorke. *Chaos: an introduction to dynamical systems*. Springer, 1996.
- [21] Lars Busing, Benjamin Schrauwen, and Robert Legenstein. *Connectivity, dynamics, and memory in reservoir computing with binary and analog neurons*. Neural Computation, 22:1272–311, 2010.
- [22] Bjorn Maes, Martin Fiers, and Peter Bienstman. *Self-pulsing and chaos in series of coupled nonlinear micro-cavities*. Physical Review B11, 7911(111), 200911.
- [23] Kristof Vandoorne, Joni Dambre, David Verstraeten, Benjamin Schrauwen, and Peter Bienstman. *Parallel reservoir computing using optical amplifiers*. IEEE Transactions On Neural Networks, 22(9):1469–1481, 2011.

6

Generating periodic patterns

A periodic pattern can be as easy as $\sin(x)$. But it can also be extremely complex, such as with robot locomotion. Imagine a walking human-like robot, which has to control over a hundred actuators, where the actuators represent a simplified set of the human muscles. What kind of algorithm can you think of to steer these inputs in such a way that it will make the robot walk? We could record the motion of a human and use that as input for the actuators. But then suppose one perturbs the robot. How will it restore its balance? Although, at first sight, the problem of generating complex motions seems very difficult to solve, it actually becomes very easy to solve by using novel on-line learning methods such as the FORCE learning method.

The FORCE learning method which was first introduced by D. Sussillo in 2009 [1], and is explained in section 2.3.4.1. This learning method, apart from the fact that that it is useful in many applications (such as robot locomotion [2–4], cognitive processing [5] and predicting chaotic attractors of dynamical systems [6]), is also very interesting from a theoretical point of view. It shows us how initially chaotic systems with complex feedback mechanisms can be stabilized. In all of the applications mentioned in [1], a leaky hyperbolic tangent reservoir was used. We would like to harness the power of photonics to bring

the performance of these systems to a new level. Furthermore, there are other, more photonic-oriented applications that could benefit from this approach. For example, generating optical patterns that can be used for data communication, optical memory, and optical switching are all possible applications where the FORCE rule could be impactful. Furthermore it is interesting to see how this theory applies to chaotic systems based on coupled resonators.

In contrast to previous work, we evaluate the proposed physical reservoir in continuous-time, i.e., without the external clock determined by the sampling period. Since periodic signals are inherently continuous, we can now fully evaluate the potential of continuous-time reservoirs.

In this chapter we will use a task that is often found in literature: the Multiple Superimposed Oscillator (MSO) problem [7]. Here, the reservoir is used to predict the evolution of a superposition of two or more sinusoidal waves with different and harmonically unrelated frequencies.

The goal of this chapter is to assess, using simulations, the use of the FORCE learning technique in an optical system and to find appropriate design parameters.

This chapter is structured as follows: we begin by describing the task (section 6.1) and by explaining how we measure the performance (the construction of the reservoir was discussed in section 4.6). In several intermediate steps we increase the complexity from a discrete-time hyperbolic tangent reservoir to a continuous-time complex-valued photonic crystal cavity reservoir. As a first intermediate step and very different to previous approaches, we will simulate a reservoir in continuous-time (i.e., with no internal clock) using a continuous-time task, which is done in section 6.3.

The second big difference between classical reservoir computing and nanophotonic reservoir computing is that the signals and states are complex-valued in the latter case. We have already pointed out that the fact that this doubles the dimensionality of the state space is beneficial for the performance on the speech task. We investigate the influence of complex-valued signals for the present task in section 6.4.

In section 6.5 we show the results of simulations for the optical reservoir using photonic crystal cavities. We discuss which parameters are relevant in building this reservoir, such as the topology, the phase difference between two resonators, the number of nodes that are biased and the delay between the nodes. A set of parameters is provided for which an optical reservoir performs particularly well on the MSO task and even outperforms the classical hyperbolic tangent reservoir. Finally, in section 6.6, we link our results to the total information processing capacity of the photonic crystal cavity reservoir for the most relevant parameters, in which the same trends are observed, i.e., the memory capacity is higher in the same regions where the MSO task performs well.

In section 6.7 we describe additional experimental challenges, such as the feedback loop, the readout layer and the weight recalculation, and we conclude in section 6.9.

Similar to the previous chapter, all simulations were performed using a combination of OGER (the OrGanic Environment for Reservoir computing [8]) and Caphe, the framework which we introduced in Chapter 4.

6.1 Task description

A nanophotonic reservoir is inherently a continuous-time system. Hence, it is appropriate to choose a continuous-time task as already mentioned. We will use the MSO task [7], a pattern generation task defined in continuous time. This academic task has previously been used to benchmark the performance of reservoirs, using several learning methods. However, the reservoir until now was always a discrete-time system, and the MSO signal was sampled at fixed timesteps.

In the MSO task, the system has to generate a superposition of sine waves with harmonically unrelated frequencies:

$$s(t) = \sin(\omega_1 t) + \sin(\omega_2 t) \quad (6.1)$$

The pulsation of the signals are: $\omega_1 = 0.2/s$ and $\omega_2 = 0.311/s$, and the target signal is sampled with timestep Δt , i.e., $s[k] = s(k\Delta t)$, $k \in \mathbb{N}$. The period of the first signal is $T_1 = 2\pi/\omega_1 \approx 31.42 s$, for the second signal this is $T_2 = 2\pi/\omega_2 \approx 20.20 s$. For a classical discrete-time reservoir, $\Delta t = 1 s$. The period of the superimposed signal is very long, which increases the challenge of learning the signal.

FORCE learning is typically used in a training and evaluation setup consisting of the following steps (also depicted in Figure 6.1):

1. Warmup ($15T_1$): the initial state $\mathbf{x}[0]$ is chosen zero. The input during the warmup is noise, sampled from a standard normal distribution with amplitude 1.
2. Training ($400T_1$): the output weights are adjusted using the proposed RLS rule [1], as we have explained in section 2.3.4.
3. Free-run ($2000T_1$): the output weights are unmodified. If the training converges, the RC system with feedback can now autonomously generate the desired function $s[k]$.

In this chapter we will approximate the continuous-time reservoir by choosing a very small simulation time step Δt , or by using an adaptive stepsize algorithm. In the latter case, a predefined accuracy has to be maintained in order

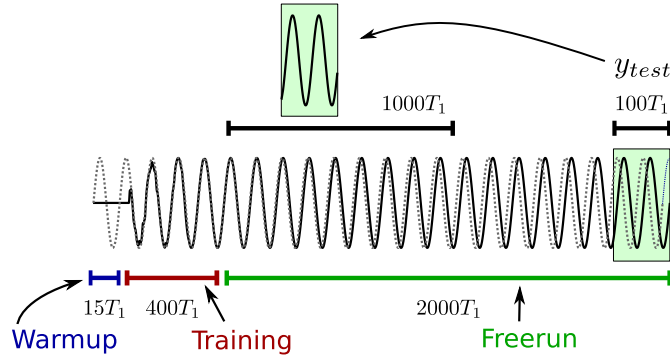


Figure 6.1: Simplified illustration of the learning sequence. T_1 is the time period of the slowest varying frequency. During warmup, the input of the reservoir is noise, sampled from a uniform distribution. During training, the output weights \mathbf{W}_{out} are adjusted such that the output (black solid line) follows the target signal (gray dashed line). The output weights are unmodified during freerun. The output can have a slightly different frequency than the target. The last samples $y_{test}[k]$ are scrolled over a window of the freerun output, each time calculating the NRMSE. The optimal value of the NRMSE is used as performance for this learning sequence.

to advance through time. The time step is controlled by estimating how close the signal with integration step Δt is to the theoretical limit where Δt converges to zero, and maintaining this value below the required accuracy. The error estimate depends on the used integration method [9].

6.2 Performance measure

The reservoir performance is evaluated through the root mean square error (NRMSE) between the output and the target function at the sampled times $k\Delta t$. In signal generation tasks, an important aspect of performance is the stability of the generated output over longer periods of time. Therefore, after the training phase, the reservoir runs for some time ($2315T_1$) before calculating the performance. Also, for periodic signals, a small phase shift between the actual and the desired output is usually acceptable, but it strongly affects the NRMSE. We use the approach of [10], where we calculate the NRMSE for windows of the output signal $y_{test}[k] = y[2315T_1 + k]$, $k \in [0, 100T_1]$, and sliding these windows over a sufficiently long section ($1000T_1$) of the free-run stage. Selecting the minimal value across all window positions effectively evaluates the shape of the desired output, largely cancelling out the impact of any phase shifts. This is illustrated

Name	Description	Default value
N	Number of neurons	200
SR	Spectral radius (see eq. (2.6))	1.5
f	Input scaling, which determines the strength of the feedback	1.0
α	Learning rate (smaller value means faster learning)	0.01
τ_0	Dominant time constant of the neuron	[2, 7] s

Table 6.1: Default values used in the leaky hyperbolic tangent reservoir.

in Figure 6.1. Although computationally intensive, this calculation can be sped up using a convolution. More details about the calculation can be found in appendix A.

Table 6.1 shows the default parameters that are used throughout this chapter for the leaky hyperbolic tangent reservoir. Any deviations from these parameters will be explicitly mentioned. We use a spectral radius of 1.5, which means the reservoir is initially chaotic¹. We simulate the continuous-time hyperbolic tangent reservoir using the Bulirsch-Stoer integration method with variable stepsize $\Delta t'$, and using a relative accuracy of 10^{-8} . Learning is still performed at discrete timesteps of $\Delta t = 1$ s.

6.3 From discrete time to continuous time

In this section we investigate the effect of the transition from a discrete-time to a continuous-time reservoir.

In literature, the MSO task is usually solved using a discrete-time reservoir. Here, the target signal is sampled using a sampling step $\Delta t = 1$ s. This means that 1 second corresponds to one step of the reservoir.

We can now also interpret this system as a continuous-time system, which is sampled every second. The feedback loop is also simulated in continuous time.

First we explain how we can view the neuron update equation of a traditional ESN reservoir as the result of applying Euler integration on ordinary differential equations (ODEs). Recall that the traditional ESN reservoir has the following update equation (see section 2.3.1):

¹In his paper, D. Sussillo explains that they initially start from a chaotic reservoir [1]. We found that similar performance is also reached for nanophotonic systems that are not initially chaotic. Of course, the spectral radius should not be too low, otherwise the dynamical system does not explore the state space efficiently during training, leading to a lower performance.

$$\mathbf{x}[k+1] = (1-\eta)\mathbf{x}[k] + \eta\mathbf{f}(\mathbf{W}_{in}\mathbf{u}[k] + \mathbf{W}_{res}\mathbf{x}[k]), \quad (6.2)$$

To more closely resemble the ODE form, we now substitute η by $\frac{\Delta t}{\tau_0}$, where τ_0 is the dominant time constant of the neuron. Here, we assume $\Delta t = 1$ s. To simplify the notation, we omit the explicit argument of \mathbf{f} , instead we write $\mathbf{f}(\mathbf{v}(t))$:

$$\mathbf{x}(t + \Delta t) = (1 - \Delta t \frac{1}{\tau_0})\mathbf{x}(t) + \Delta t \frac{1}{\tau_0} \mathbf{f}(\mathbf{v}(t)).$$

Restructuring and taking the limit for $\Delta t \rightarrow 0$ yields the equivalent continuous-time leaky hyperbolic tangent ODE equations:

$$\frac{d\mathbf{x}(t)}{dt} = -\frac{1}{\tau_0}(\mathbf{x}(t) - \mathbf{f}(\mathbf{v}(t))) \quad (6.3)$$

Note that $\mathbf{v}(t)$ also needs to be adapted to explicitly include the interconnection delays in the network. In a discrete-time RC system there is inherently a delay of 1 s between the neurons. To most closely resemble this classical case, all delays are chosen equal to the original discrete time step of $\Delta t = 1$ s, i.e.:

$$\mathbf{v}(t) = \mathbf{W}_{in}\mathbf{u}(t) + \mathbf{W}_{res}\mathbf{x}(t - \Delta t).$$

In a more general setting, a delay on the input connections can also be included.

For most experiments that we perform in this chapter with hyperbolic tangent reservoirs, we sweep τ_0 , the time constant of the neuron. A smaller τ_0 corresponds to a faster neuron. The average NRMSE values are summarized in Figure 6.2, which shows that moving to continuous time yields a decrease of the NRMSE over the full parameter range. This improvement is partly bandwidth-related: certain frequencies cannot be captured in discrete time whereas in our case we effectively simulate a continuous-time system (this is done either by choosing a sufficiently fine timestep or by using an adaptive integration step).

For both reservoir types, we can identify an optimal value of the neuron time constant. For the discrete-time reservoir, we find an optimal NRMSE of 0.127 for $\tau_0 = 4.0$, with a sample standard deviation of 0.111. For the continuous-time reservoir, the best NRMSE equals (0.066 ± 0.054) , for $\tau_0 = 3.0$ s.

In Figure 6.2(b) we show the same experiment in continuous time, and add the experiment where the interconnection delay and feedback delay are equal to zero. For this continuous-time reservoir, the best NRMSE equals (0.050 ± 0.022) , which occurs at the lower limit of the interval considered (i.e. $\tau_0 = 2.0$ s). Because there is no delay in the feedback loop, the system responds faster to the changes that are made in the connections, which in general is beneficial for the performance. A zero delay is of course not physical, but such a strategy could be used to improve the performance of simulation-only reservoirs.

These results show that, for the signal generation task, a continuous-time hardware reservoir can perform better than a discrete-time reservoir, or equivalently: the same performance can be reached with fewer neurons, which is an advantage when implementing a hardware reservoir computer.

6.4 From real-valued to complex-valued reservoirs

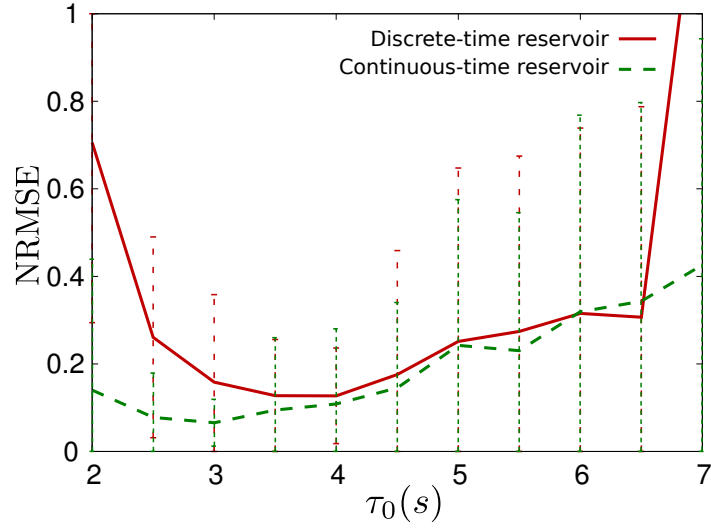
One of the advantages of using photonics for a hardware reservoir is the fact that when using coherent light sources, the optical signals are complex-valued, i.e. they have an amplitude and phase. In order to understand how this gives an advantage, we now extend the discrete-time leaky hyperbolic tangent reservoir with complex-valued states and compare this with the original real-valued reservoir. Note that in the previous chapter we have immediately used a complex-valued nanophotonic reservoir, based on our photonic crystal cavities, and not a complex-valued reservoir based on leaky hyperbolic tangent neurons. In this section, we explicitly compare a real-valued and a complex-valued reservoir for the same neuron type. The input weights are also complex-valued. The readout layer splits the states $\mathbf{x}(t)$ into a real and imaginary part, before multiplying them with \mathbf{W}_{out} .

Because the signals are now complex-valued, we need to define a new non-linearity for the neuron. It is tempting to replace the hyperbolic tangent function by $f(z) = \tanh(z)$, $z \in \mathbb{C}$. However, this function has discontinuities at $f(j(2k+1)\pi/2)$, $k \in \mathbb{Z}$ which are generally not wanted. We want to guarantee continuity and keep the $\tanh(x)$ behavior with an image that is bound to $[-1, 1]$. We can do so by preserving the phase of the signal and applying \tanh on the absolute value of the signal:

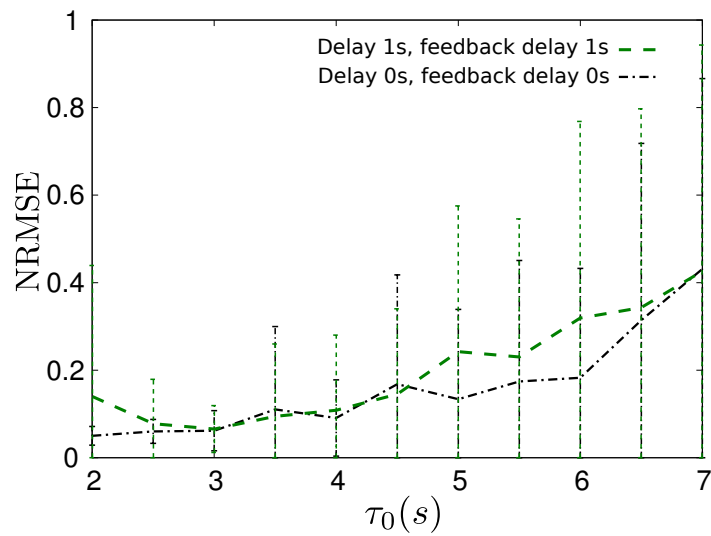
$$f(z) = e^{-j\angle(z)} \tanh(|z|), \quad (6.4)$$

where $\angle(z)$ is the angle of z .

Figure 6.3. shows the performance of the complex-valued reservoir. Because the state space of a complex-valued reservoir contains twice as many variables as that of a real-valued reservoir, the performance has improved. Also, the performance of the MSO task for the complex-valued reservoir is similar to the performance of a real-valued reservoir with twice as many nodes. This is also shown in Figure 6.3. Recall that for the discrete-time case, the optimal NRMSE value was 0.127 ± 0.111 . For $\Delta t = 1.0$ s, the optimal value is found for $\tau_0 = 4.5$ s. For the same network, but with complex-valued states, the optimal value is found for $\tau_0 = 3$ s, and the corresponding NRMSE equals 0.0546 ± 0.0238 . We also compare this to a real-valued network of with twice the number of neurons, also shown in Figure 6.3. The performance is similar: the best NRMSE of



(a)



(b)

Figure 6.2: Normalized Root Mean Square Error (NRMSE) for different τ_0 for the MSO task described in section 6.1. The error bars show the sample standard deviation over 40 simulations ($NRMSE \pm \sigma_{NRMSE}$), and the dominant time constant of the neurons (τ_0) is swept. Top: the performance of the continuous-time reservoir (dotted green) is better than that of the classical reservoir (red). Bottom: Without delay between the neurons or in the feedback loop, the reservoir can respond faster to changes in the output, leading to a better performance.

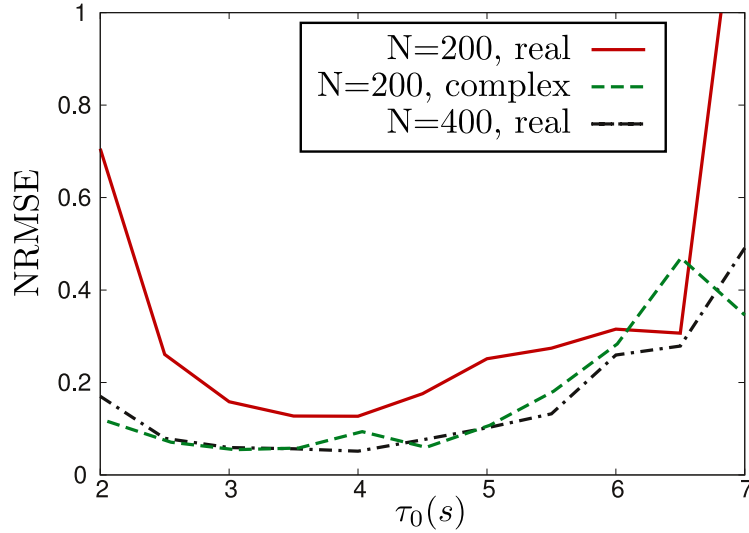


Figure 6.3: The NRMSE for three reservoirs as a function of the leak rate: standard leaky hyperbolic tangent reservoir with 200 neurons (red, baseline), the same reservoir with complex-valued states (green, dashed) and a standard reservoir with 400 neurons. Clearly, the complex-valued reservoir performs better than the standard reservoir. For most leak rates, the performance is similar to the system with 400 neurons.

0.0514 ± 0.0233 is found for $\tau_0 = 4$ s. We can conclude that, for this task, the performance of a coherent (i.e., complex-valued) reservoir approximates that of a real-valued reservoir with twice the number of neurons.

6.4.1 Continuous and complex-valued

In this section we investigate the effect of using both a continuous time and complex-valued reservoir, which combines both effects which appeared to be beneficial for the performance of the reservoir. In Figure 6.4, we show the results of the previous sections, and add the experiment with a complex-valued, continuous-time reservoir (the error bars are removed for clarity). As can be seen from the figure, the improvement due to using complex variables is dominant, whereas the additional improvement due to continuous time is not significant. Only in the region with small τ_0 , the performance is slightly better. The best performance occurs at the lower limit of the interval considered, and equals 0.051 ± 0.024 .

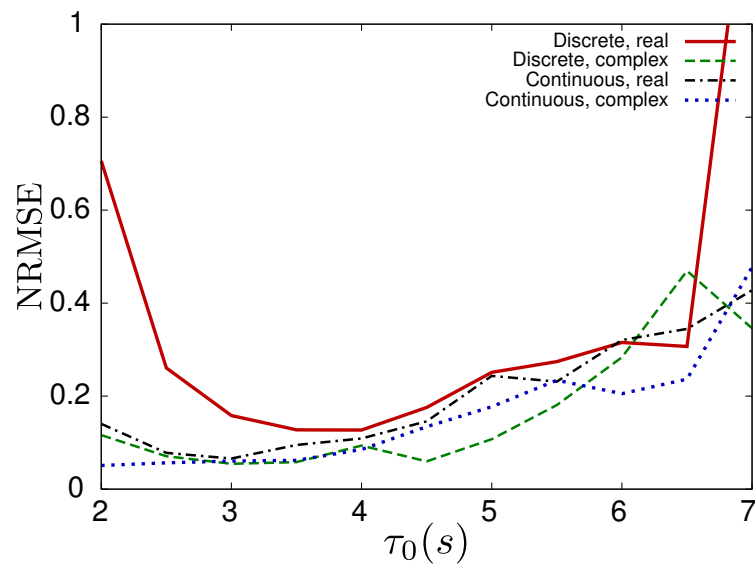


Figure 6.4: Comparison of discrete time and continuous time reservoirs, both for with real-valued and for complex-valued neurons. The performance gain when using both using complex-valued states and a continuous-time reservoir is not significant compared to complex-valued neurons in a discrete-time reservoir or real-valued neurons in a continuous-time reservoir.

Name	Description	Default value
width	Width of the 2D mesh	20
height	Height of the 2D mesh	10
N	Number of cavities	200
BF	Bias fraction (fraction of nodes that receive bias)	0.1
P_{bias}	Bias power	$1.30P_0$
ϕ_j	Phase reflection of the cavities	0.2π
FB	The strength of the feedback	1.0
λ	Input wavelength	1551.83 nm
λ_r	Cavity resonance wavelength	$1550 \text{ nm} \pm 0.2 \text{ nm}$
τ	Cavity lifetime	1.39 ps
Δ	Detuning of the cavity ($= \tau(\omega_r - \omega)$)	2
f_1	Target signal frequency	$0.3/\tau \simeq 216 \text{ GHz}$
α	Learning rate (smaller: faster)	0.01
τ_d	Delay between the resonators	0 ps

Table 6.2: Default values used in the photonic crystal cavity reservoir.

6.5 The full Photonic Crystal Cavity reservoir

In this section we discuss the results of the MSO task using the full Photonic Crystal Cavity (PhCC) reservoir, i.e., operating in the complex regime in continuous time, and with the dynamical behavior of the photonic crystal cavities. The reservoir parameters are summarized in Table 6.2. The target signal is again given by the sum of two sines:

$$s(t) = \sin(2\pi f_1 t) + \sin(2\pi f_2 t) \quad (6.5)$$

Here, $f_1 = 0.3/\tau \simeq 216 \text{ GHz}$ in order to fully exploit the internal dynamics of the nodes. We then set $f_2 = 0.311/0.2 f_1$, so the ratio of frequencies of the original task is preserved.

We will discuss the influence of several important design parameters for the photonic crystal cavity reservoir, such as the phase between the resonators, the delay in the waveguides connecting the cavities, the splitting ratio S (see Figure 4.12) and the network size. In many cases it is instructive to use the same parameters in a small system of two coupled cavities, as we did in chapter 3. This will allow us to understand the results more intuitively.

6.5.1 Phase reflection of the resonator

The phase reflection ϕ_j strongly influences the dynamics of the optical resonator. The total phase difference between two connected resonators equals $\phi_i/2 + \phi_j/2 + \phi_{i,j}$, where $\phi_{i,j}$ is the additional phase propagation, caused by the splitters and waveguides connecting the two resonators. We will keep $\phi_{i,j}$ fixed and only modify the ϕ_i of the cavities.

For a dynamical system, a linear stability analysis reveals whether the un-driven system is stable or not. This is done by examining the eigenvalues of the Jacobian of the system. If all eigenvalues have a negative real part, the system is stable, as we have seen in the previous chapter. In some cases an unstable fixed point implies chaos or self-pulsation. To distinguish between both, one can calculate the maximal Lyapunov exponent of the system. If the maximal Lyapunov exponent is larger than zero, the system is chaotic. For a stable periodic solution, the maximal Lyapunov exponent is zero.

This is elaborated in detail in [11]. For example, a series of two resonators will self-pulsate when $\phi_j \simeq 0.2\pi$, $P_{in} \simeq P_0$ and $\Delta = 2$. For larger circuits with arbitrary topology, it becomes very cumbersome to evaluate the Jacobian and calculate the largest Lyapunov exponent. Numerical simulations of an optical reservoir, with parameters used from Table 6.2, show that the reservoir is indeed in a region where self-pulsation occurs. We chose $\phi = 0.2\pi$ (the same phase as we chose in chapter 3) and feed an input power $P_{bias} = 1.3P_0$ to a fraction (BF in Table 6.2) of the cavities. When this fraction is sufficiently large ($>10\%$), enough power arrives in the cavities and self-pulsation occurs. Under these conditions, the training and/or free-run are disturbed severely and most of the time, training does not converge.

This indicates that preferentially, there should be no self-pulsation or strong synchronized interaction between the cavities. To understand this behavior more quantitatively, we change the phase reflection to $\phi_j = 0.2\pi + \phi_r \epsilon$, where $\epsilon \sim \mathcal{N}(0, 1)$ is sampled from a Gaussian random variable with zero mean and 1 variance. Physically, this can be done by changing the length of the interconnection between the different cavities. ϕ_r is thus the amount of phase randomness that we add to the resonators. When increasing this value, we move out of the self-pulsing regime for the case of two cavities, which is an indication that the interaction between the cavities is less synchronized. Indeed, also for the full reservoir the performance improves. This can be seen in Figure 6.5, where we sweep ϕ_r between 0-2. We have also performed the simulation for different fractions BF of the nodes receiving bias input. The more cavities that receive bias, the more the reservoir dynamics are disturbed by strong interactions between the resonators. As a consequence, the training is more difficult, which leads to a decreased performance, or even the inability to reproduce the target signals. With no bias and randomized phases, an NRMSE of 0.030 ± 0.021 is found. This

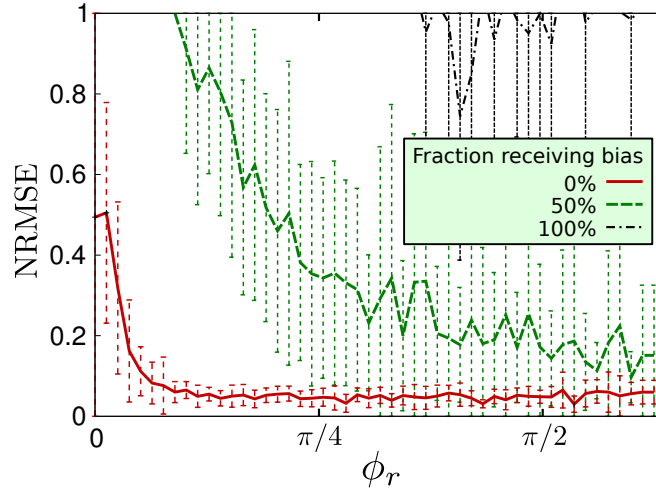


Figure 6.5: The error (NRMSE) after training an optical network of photonic crystal cavities for the MSO task. The phase between the resonators is described by $\phi_j = 0.2\pi + \phi_r \epsilon$, $\epsilon \sim \mathcal{N}(0, 1)$. This is done for different fractions of cavities receiving bias. The more cavities that receive bias, the more the reservoir dynamics are disturbed by strong interactions between the resonators (e.g. self-pulsation), which causes the network to be unable to generate the signal autonomously. Increasing the randomness in the phase reduces the amount of self-pulsation.

is the best value which we found for this chapter, and better than the classical hyperbolic tangent reservoir.

On the silicon photonics platform [12], the phase errors produced by propagation over a photonic wire are very small. Over a few $100 \mu\text{m}$, the phase error for two identically designed waveguides is less than 0.1π [13], and the technology is improving steadily. By changing the length or width of the waveguide, we can change this relative phase.

6.5.2 Delays

Although the phase and the delay of a waveguide with length L both scale linearly with L , it is more convenient to investigate the effects separately. This is justified by the fact that one only needs $0.5 \mu\text{m}$ length to rotate the phase by 1π , while one needs at least $50 \mu\text{m}$ of waveguide length to see a significant delay (of approximately 0.1 ps). Again, we first look at the dynamics of a sequence of two coupled cavities. The response for different delays for this system is shown in Figure 6.6. Here, the delay is very important for the condition for self-pulsation and the shape of the output signal. For $\phi = 0.2\pi$, $P_{in} = 1.30P_0$ and $\delta = -2$, and

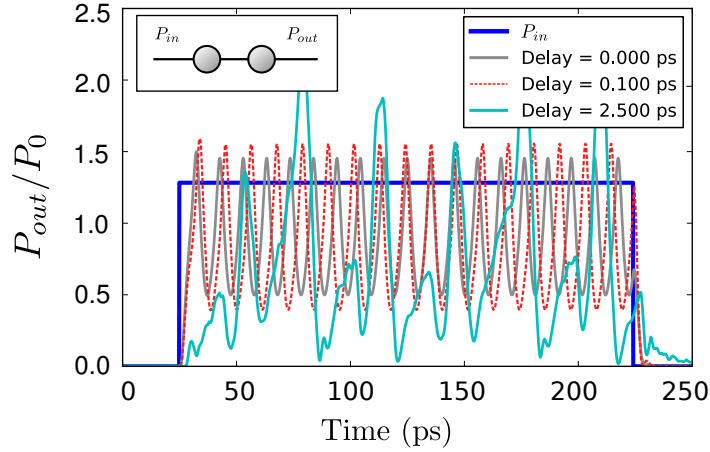


Figure 6.6: Dynamics for a sequence of two coupled resonators (shown in the inset). With increasing delay, the dynamics change from self-pulsation to chaos. A delay of 0.1 ps corresponds to approx. $10 \mu m$ on-chip.

for a delay larger than $2.5 ps$ (which corresponds to approximately $310 \mu m$ on-chip), the self-pulsation is lost and the output becomes chaotic. The influence of the delay for the MSO task is summarized in Figure 6.7. Where the resonators were initially locked in a self-pulsing regime (which caused learning to fail), increasing the delay now improves the ability to learn. We only simulate up to $1.0 ps$ as the trend doesn't change anymore after this delay. If the system was initially capable of learning (e.g. when the phases were randomized), increasing the delay has almost no influence on the performance of the reservoir. For $0.8 ps$ and a bias fraction of 10% the NRMSE is 0.024 ± 0.028 .

6.5.3 Splitters

For all simulations until now, the splitting ratio S (see Figure 4.12 and Figure 6.8) was neglected, and we measured the power of the cavities without disturbing the system. Also, the input was fed to the cavities without introducing extra loss. In practice however, when feeding power to a cavity, this has the drawback that power also leaks away through the same port. For example, a splitting ratio of $10/90$ ($S = 0.10$) means that 10% of the source power reaches the cavity, but it also means that 10% of the cavity power leaks away through this channel. 90% of the power circulates to other cavities. To be able to draw meaningful comparisons between different values of S , the input signal amplitude and feedback strength have been scaled by $1/\sqrt{S}$. The splitting ratio should be kept small to ensure that enough power circulates in the reservoir. This of course means that

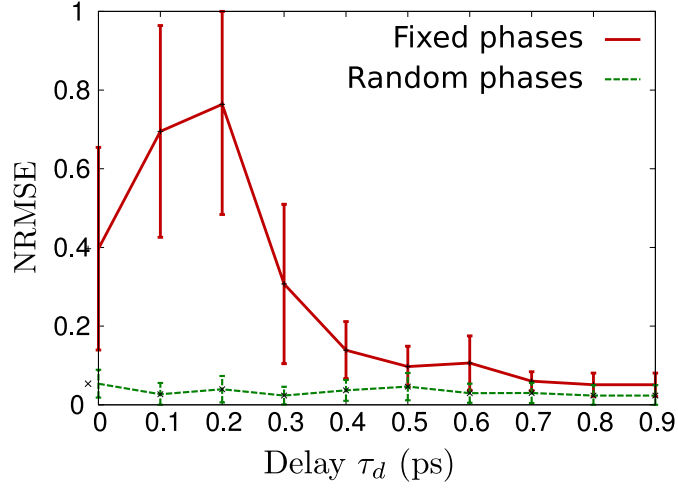


Figure 6.7: Error (NRMSE) of the photonic reservoir for the MSO task. As shown in Figure 6.5, training fails in certain conditions when the phases are fixed and equal to 0.2π . By increasing the delay (100 ps delay is approximately $12.5 \mu m$ on-chip), the self-pulsation in a series of two cavities is lost (see Figure 6.6). The conclusion that the training works better when self-pulsing is not present is also valid here.

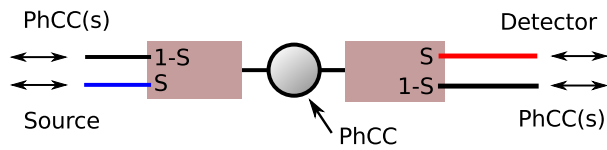


Figure 6.8: Illustration of the splitting ratio S . Additional splitters are needed in order to send the signal from the source into the photonic crystal cavity, and from the cavity to the detector.

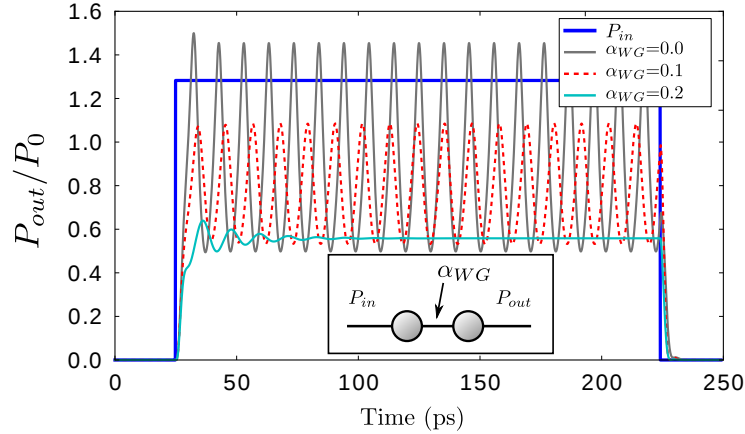


Figure 6.9: Dynamics of a series of two coupled resonators for increasing attenuation of the waveguide (α_{WG}) between the cavities. α_{WG} is the one-way power attenuation. For $\alpha_{WG} = 0.2$, the self-pulsation is lost.

higher input powers are required.

Again, studying the dynamics of a system of two coupled resonators gives insight about the behavior of the reservoir. In this case, we increase the attenuation of the waveguide between the two resonators, which has the same effect as splitting some of the power in the full reservoir. For example, when the splitting ratio S of Figure 4.12 equals 0.1, this is equal to $1 - 0.9^2 = 19\%$ power attenuation in the waveguide (two splitters added between the two resonators). For this attenuation, the interaction between two cavities reduces considerably. In Figure 6.9 one can clearly see that the self-pulsation is lost when the attenuation becomes bigger.

The influence of the splitter ratio on the training is shown in Figure 6.10. When initially the training fails because the bias fraction is large, causing strong unwanted interactions between the cavities (for example self-pulsation), increasing the splitting ratio will decrease these strong interactions between the neurons which improves the results. In the other case, where the bias fraction is low (and initially the training converges), there is no significant influence of this splitting ratio on the training (although the error increases slightly for larger S). In our experiments, the bias input powers and the feedback strength were scaled by a factor of $1/\sqrt{S}$. A larger S means that less external power needs to be added (bias) or amplified (feedback connection).

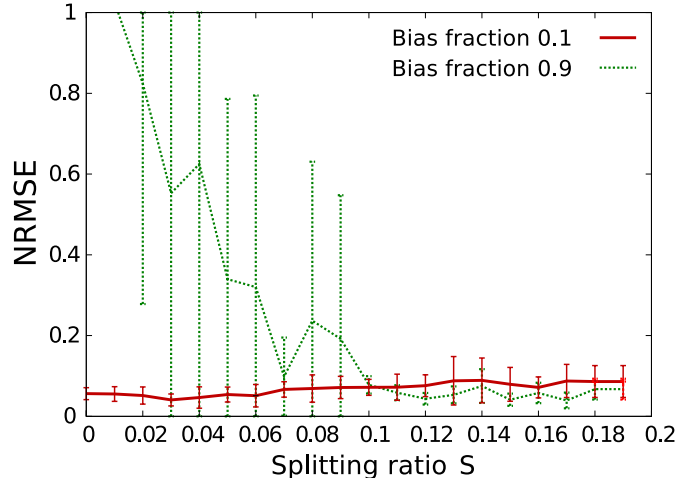


Figure 6.10: Influence of the splitter ratio S (see Figure 4.12 and Figure 6.8). By increasing the splitting ratio, the interaction between two neighboring cavities is decreased. This is advantageous for learning, because the strong self-pulsation which disrupts training disappear (see also Figure 6.9). Parameters: phases random, $P_{bias}=1.3P_0/\sqrt{S}$, $FB = 1.0/\sqrt{S}$.

6.5.4 Restricting the readout to the power only

In all previous results, we have read out the real and imaginary part of the neurons separately. This means that the readout has twice as many inputs as the number of neurons in the reservoir. We can also explicitly read out the squared magnitude of the signals that arrive at the readout layer. This corresponds to a more realistic scenario in case we have to convert the signal to the electrical domain for reading out the states², using for example an integrated photodiode.

In Figure 6.11, we show the performance of the reservoir as a function of the fraction of neurons that receive bias input ($P_{bias} = 1.3P_0$) for both cases. When the neurons receive no bias, the reservoir is closer to the linear regime. In this case, when we read out the power of the neurons (magnitude squared), the performance is very bad. This is due to the fact that the frequency of a periodic signal is doubled by the squared readout. Suppose the system has to learn a signal proportional to $\sin(\omega_1 t)$. In the readout layer, because of the squaring, there are only higher-order frequency terms available ($2\sin^2(\omega_1 t) = (1 - \cos(2\omega_1 t))$), while the system should be trained to reproduce the original

²Ideally however, we would like the system to be all-optical. In this case, we would have a complex readout with N variables and a complex-valued target signal, a situation which has not been studied in this work.

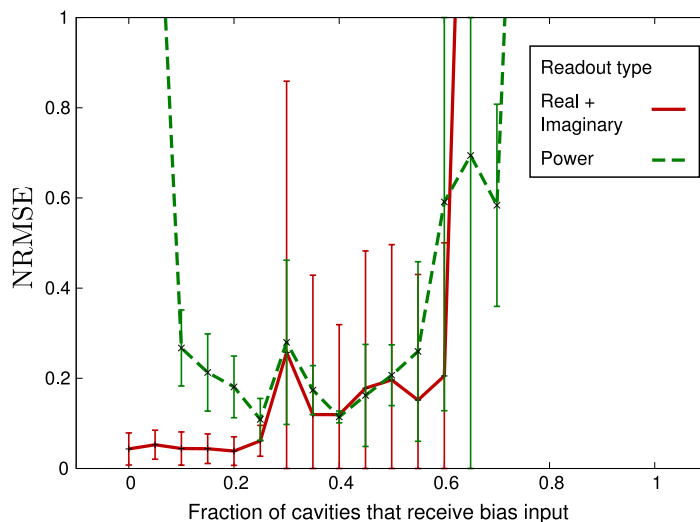


Figure 6.11: Comparison of reading out the the real and imaginary part (as we have done in previous experiments), or reading out the power of the reservoir.

signal $\sin(\omega_1 t)$. When a bias term is added, the original frequency terms are still present $((a + \sin(\omega_1 t))^2 = a^2 + 2a \sin(\omega_1 t) + \sin^2(\omega_1 t))$, allowing the system to be trained efficiently.

As can be seen in Figure 6.11, adding bias improves the performance up to a certain point. As we saw previously in Figure 6.5, increasing the bias too much will drive them again into a regime with too much nonlinearity, where the reservoir cannot perform properly.

6.5.5 Resonance frequency changes due to fabrication imperfections

Variations caused by fabrication imperfections cause the resonance frequencies of optical resonators to be different from one device to the other. This shift can be up to 1 nm [12] for ring resonators³. For our simulations, we used a variation on the resonance of 0.2 nm (see Table 6.2). Above 0.3 nm, some of the simulations do not converge, and above 0.4 nm, no reservoir succeeds at reproducing the MSO signal. As the technology improves, this resonance shift due to fabrication errors is becoming increasingly smaller. Conceptually, one can compensate

³Ring resonators are used more often than photonic crystal cavities, and a lot of effort is being done to reduce their variability. In our measurements from section 3.3, we found a variation of 5 nm for photonic crystal cavities, but these are not for identical cavities that are closely together, but for identical cavities on different dies. We have not measured identical cavities that are closely together.

the individual resonances either by trimming the individual devices [14] or adjusting the temperature of each device with heaters [15, 16]. But also alternative designs can relax the requirements on the variability of the system. For example, a smaller cavity lifetime means that the resonance is less pronounced, making it less susceptible to process variations, but requiring higher input powers to achieve nonlinear behavior.

6.5.6 Network size

For the final MSO experiment, we measure the performance of the network as a function of the network size. The actual fabricated layout could have a rectangular or square shape. For that reason, we perform the experiments for two slightly different topologies: a square mesh and a rectangular mesh. From Figure 6.12 we can conclude that for the chosen target signal $s(t)$ (see equation 6.5), using more than 70 resonators does not further improve the performance⁴. Furthermore, the difference between a rectangular and square topology is negligible for this task.

Due to the increased performance of the continuous time, complex-valued photonic reservoir compared to the classical hyperbolic tangent reservoir, typically less neurons will be needed to obtain the same performance.

6.6 Information processing capacity

In [17], the *(linear) memory capacity* of reservoirs was introduced. It quantifies a reservoir's capacity to reproduce past input samples in a task-independent way. For many tasks, a larger linear memory capacity will result in better performance of the reservoir. In [18], this concept was generalized to the *total information processing capacity*. This measure quantifies the total capacity of a dynamical system to compute transformations, both linear and nonlinear, of its input history. This total capacity is bounded by the number of observed state variables. In the case of our photonic reservoir, when using the N powers as readout, this equals the number of resonators in the system. This maximal capacity is effectively reached for dynamical systems with fading memory, i.e., that are asymptotically stable during their entire driven operation. Although the experimental quantification of the total capacity can be a bit tedious, reasonable approximations of the total capacity can often be achieved within an acceptable computation time. In practice, the useable capacity decreases rapidly in the presence of noise if the responses of the reservoir states to the input are very

⁴In classical reservoir computing, the network size is usually around 1000 neurons. As this is clearly not realistic for a first hardware implementation, we initially compared the hyperbolic tangent and nanophotonic reservoir for 200 neurons.

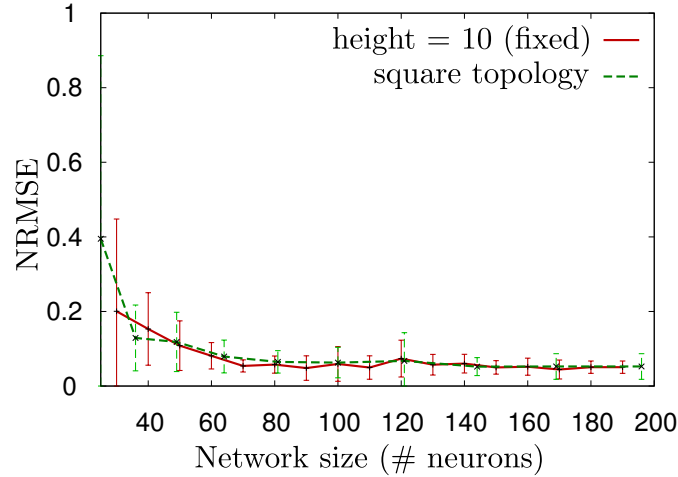


Figure 6.12: Influence of the network size on the performance. We have simulated two variations on the mesh topology, once with a square topology and once with a rectangular topology. Clearly the influence of this slight topology change is negligible. It also shows that, for this specific task, 70 resonators are sufficient, and there is no performance gain by using more resonators.

similar. In order to perform well as a reservoir, a dynamical system should first achieve close to its maximal capacity, i.e., be stable and have sufficient variability in its responses to the input signals.

We will now apply this knowledge to a small reservoir⁵ of 25 photonic crystal cavities, and consider two important parameters: the input scaling and the phase randomness. The amount of nonlinearity is determined by the power inside a cavity (and hence by the input scaling), so this will have a large influence on the dynamics of the network. Also the phase randomness in the phase reflection ϕ_r will greatly influence the performance (as we saw in Figure 6.5). The other reservoir parameters are fixed, and are taken from Table 6.2. In this experiment we have used the magnitude of the output of the neurons instead of using the real and imaginary parts separately. As can be seen from Figure 6.13, the total information processing capacity (with a maximum of 25) drops dramatically when the input scaling increases and there are no random phases. This again corresponds to the self-pulsation regime which we encountered in the previous sections. This means that for typical RC tasks, the performance will be better when not in a self-pulsing regime.

⁵Note that for this experiment, as opposed to the other experiments in this chapter, we do not include the feedback loop as shown in Figure 2.7. For this experiment we use a normal reservoir as shown in Figure 2.5.

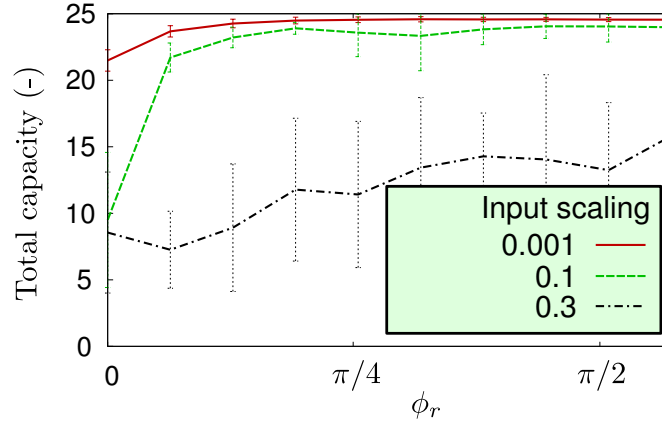


Figure 6.13: Total information processing capacity of a 5x5 photonic crystal cavity network. The region with low phase randomness and high input powers correspond with self-pulsation regions. These regions are better avoided in order to increase the total capacity, and hence the performance of the system. This conclusion is in line with previous experiments.

6.7 Hardware challenges

When moving to a future hardware implementation, there are two extra challenges to be solved: the feedback loop and the readout. Conceptually, it is very easy to make the feedback loop all-optical: the output weights can be modified externally, for example by using a Mach Zehnder Interferometer (MZI) and applying a voltage in one of the arms. This causes constructive/destructive interference, which allows us to modify the readout weight. An amplifier is needed to amplify the output before feeding it back to the input. This can be implemented by designing an SOA in the feedback loop. For example, the amplifier from [19] can be used, which is demonstrated on the same silicon photonics platform.

Another challenge is reading out the states and recalculating the weights in a way that is fast enough: the outputs have to be read out, fed to a computer to calculate the weight adjustment, and then the output weights have to be modulated. Other learning techniques can simplify the calculations, such as a more simple delta-type rule in which we do not need to calculate a P matrix (also shown in [1]), or by using Hebbian learning [20], where the exact error signal is not explicitly needed.

For this theoretical study, which builds upon the results of [11, 21], free carrier effects (\sim ns) and temperature effects ($\sim 0.100 \mu\text{s}$) have not been taken into account. However, the CMT equations and steady-state curves for these type of

dynamical effects are very similar, which means that it should not be difficult to find good working regimes for the reservoir in case the other effects are taken into account.

Depending on the specific design parameters and the material system, some of the effects will be more important than other effects: due to the nonlinear two photon absorption, the free carrier effects in Silicon on Insulator at $\lambda = 1.55\mu m$ are not negligible. Also, due to the high powers inside the cavities, the temperature will have a significant effect, albeit on a slow timescale compared to the fast Kerr effect. Moving to different material systems or different wavelengths can change these conditions.

6.8 Choosing phenomenological parameters instead of physical parameters

In all research we have performed in this and the previous chapter, the main focus has been on investigating the influence of phenomenological parameters on the reservoir rather than using physical parameters. For example, we chose to sweep the phase of a waveguide rather than sweeping the physical length and/or the effective index of the waveguide. Or, we define a phase reflection ϕ_j rather than taking into account the actual dimensions of the photonic crystal resonator. In our simulation results, we also fully decoupled the phase and delay, even though the two depend on each other. This does not really impose a restriction in reality, as for a small length difference, we can cover all phases $[0, 2\pi]$, while the time delay does not change significantly. With a length difference of λ/n_{eff} ($\approx 400nm$) we can cover all phases, while this corresponds to a negligible delay of $\lambda/n_{eff}/c$ ($\approx 1.3fs$).

This allows us to decouple the analysis of the nanophotonic reservoir from the actual physical layout, and allows us to generalize many of the results.

Later, once a good set of phenomenological parameters are found, a physical geometry can be derived based on experimental results.

6.9 Conclusion

In this chapter we have studied different reservoir architectures for the generation of periodic patterns. The output weights are trained using an online technique called FORCE (see section 2.3.4.1). The benchmark task we used is the standard MSO task, and the performance is measured by the normalized root mean square error (NRMSE). Table 6.3 summarizes the result of all simulations performed in this paper.

Reservoir	Time	States	comments	NRMSE
Hyperbolic tangent	DT	real		0.127 ± 0.111
	CT	real		0.066 ± 0.054
		complex	No delays	0.050 ± 0.022
	DT	complex		0.0546 ± 0.029
	DT	real	Double size (N=400)	0.0514 ± 0.0233
Photonic crystal cavities	CT	complex	Randomized phase, no bias fraction	0.030 ± 0.021
		complex	Randomized phase, 10% nodes biased, 0.8 ps delay	0.024 ± 0.028

Table 6.3: Summary of the Normalized Root Mean Square Errors (NRMSE) calculated in this chapter for the different architectures. DT = discrete time, CT = continuous time. Tanh: classical hyperbolic tangent neurons, PhCC: photonic crystal cavity.

Unlike standard reservoir computing, where a discrete-time system is used, we use advanced integration routines to simulate a reservoir in continuous time, and find that the reservoir performs better for the MSO task, which is inherently a continuous-time task. Also, using a complex-valued reservoir improves the general performance because of the increased state space.

We presented a hardware implementation for reservoir computing using photonic crystal cavities. These resonators exhibit bistability because of the Kerr nonlinearity, and they are modeled using Coupled Mode Theory (CMT), as we have shown in section 3.2.1.

There are several important design parameters such as the topology, the phase difference between the cavities and the delay between the cavities. We show that it is important not to drive the cavities in a self-pulsating regime, because strong interaction between neighboring resonators disturbs the training process and decreases the final performance. After optimizing the parameters of the optical reservoir, we find that it outperforms the classical hyperbolic tangent reservoir: the average NRMSE is 0.024 compared to an average NRMSE of 0.127 for the hyperbolic tangent reservoir. It also outperforms the complex-valued continuous time hyperbolic tangent systems, which shows that the dynamics of the cavity can be an extra factor which improves the performance. This conceptual study shows that nanophotonic structures such as the photonic crystal cavities are a good candidate for generating periodic patterns in the optical domain. There are however some challenges to overcome to create an all-optical hardware implementation: the readout needs to be computed

fast enough, many data signals need to be provided on-chip, the variation on the resonances frequencies of the photonic crystal cavities should be small enough, and the feedback signal needs to be strong enough.

References

- [1] David Sussillo and L. F. Abbott. *Generating Coherent Patterns of Activity from Chaotic Neural Networks*. *Neuron*, 63(4):544–557, AUG 27 2009.
- [2] RC Miall, DJ Weir, DM Wolpert, and JF Stein. *Is the cerebellum a Smith predictor?* *Journal of motor behavior*, 25(3):203–216, 1993.
- [3] Graham W Taylor, Geoffrey E Hinton, and Sam T Roweis. *Modeling human motion using binary latent variables*. *Advances in neural information processing systems*, 19:1345, 2007.
- [4] Francis Wyffels and Benjamin Schrauwen. *Design of a central pattern generator using reservoir computing for learning human motion*. In *Advanced Technologies for Enhanced Quality of Life, 2009. AT-EQUAL'09.*, pages 118–122. IEEE, 2009.
- [5] Rafael Yuste, Jason N MacLean, Jeffrey Smith, and Anders Lansner. *The cortex as a central pattern generator*. *Nature Reviews Neuroscience*, 6(6):477–483, 2005.
- [6] Herbert Jaeger and Harald Haas. *Harnessing Nonlinearity: Predicting Chaotic Systems and Saving Energy in Wireless Communication*. *Science*, 304(5667):78–80, 2004.
- [7] Jochen J. Steil. *Several ways to solve the MSO problem*. In *ESANN*, pages 489–494, 2007.
- [8] David Verstraeten, Benjamin Schrauwen, Sander Dieleman, Philemon Brakel, Pieter Buteneers, and Dejan Pecevski. *Oger: Modular Learning Architectures For Large-Scale Sequential Processing*. *Journal of Machine Learning Research* (submitted), 2011.
- [9] William H. Press, Saul A. Teukolsky, William T. Vetterling, and Brian P. Flannery. *Numerical Recipes: The Art of Scientific Computing*. 2007.
- [10] X. Dutoit, B. Schrauwen, J. Van Campenhout, D. Stroobandt, H. Van Brussel, and M. Nuttin. *Pruning and regularization in reservoir computing*. *Neurocomputing*, 72(7-9):1534 – 1546, 2009.

- [11] Bjorn Maes, Martin Fiers, and Peter Bienstman. *Self-pulsing and chaos in series of coupled nonlinear micro-cavities*. Physical Review B11, 7911(111), 200911.
- [12] S.K. Selvaraja, W. Bogaerts, P. Dumon, D. Van Thourhout, and R. Baets. *Sub-nanometer Linewidth Uniformity in Silicon Nanophotonic Waveguide Devices Using CMOS Fabrication Technology*. Selected Topics in Quantum Electronics, IEEE Journal of, 16(1):316–324, jan.-feb. 2010.
- [13] Pieter Dumon. *Ultra-Compact Integrated Optical Filters in Silicon-on-insulator by Means of Wafer-Scale Technology*. MAR 1 2007.
- [14] J. Schrauwen, D. Van Thourhout, and R. Baets. *Trimming of silicon ring resonator by electron beam induced compaction and strain*. Optics Express, 16(6):3738–3743, 2008.
- [15] Daoxin Dai, Liu Yang, and Sailing He. *Ultrasmall Thermally Tunable Microring Resonator With a Submicrometer Heater on Si Nanowires*. Light-wave Technology, Journal of, 26(6):704–709, march15, 2008.
- [16] Ciyuan Qiu, Jie Shu, Zheng Li, Xuezhong Zhang, and Qianfan Xu. *Wavelength tracking with thermally controlled silicon resonators*. Optics Express, pages 5143–8, 2011.
- [17] H. Jaeger. *The "echo state" approach to analysing and training recurrent neural networks*. GMD Report 148, GMD - German National Research Institute for Computer Science, 2001.
- [18] Joni Dambre, David Verstraeten, Benjamin Schrauwen, and Serge Massar. *Information processing capacity of dynamical systems*. SCIENTIFIC REPORTS, 2:1–7, 2012.
- [19] M. Tassaert, G. Roelkens, H. J. S. Dorren, D. Van Thourhout, and O. Raz. *Bias-free, low power and optically driven membrane InP switch on SOI for remotely configurable photonic packet switches*. Opt. Express, 19(26):B817–B824, Dec 2011.
- [20] Robert Legenstein, Steven M Chase, Andrew B Schwartz, and Wolfgang Maass. *A reward-modulated hebbian learning rule can explain experimentally observed network reorganization in a brain control task*. Journal of Neuroscience, 30:8400–8410, 2010.
- [21] M. Soljacic, M. Ibanescu, S. G. Johnson, Y. Fink, and J. D. Joannopoulos. *Optimal Bistable Switching in Nonlinear Photonic Crystals*. Phys. Rev. E, 66:055601, 2002.

7

Conclusions and perspectives

In this chapter we summarize the work that has been done in this dissertation. We look at the future perspectives for the novel research field of nanophotonic Reservoir Computing (RC). We explain the steps that are needed in order to create a first experimental prototype, based on Photonic Crystal Cavities (PhCC). Then we explain the research steps that can be taken based on the results of this dissertation.

7.1 Summary

This dissertation has focused on two different aspects of nanophotonic reservoir computing. The first focus was on the modeling platform. We have designed a framework for efficient simulations of large nonlinear optical circuits (see chapter 4). It was originally developed for the application of nanophotonic reservoir computing, but it is now used in many other domains of nanophotonics as well. With the software framework that was developed during this dissertation, any integrated nanophotonic chip can be efficiently modeled at the circuit level¹, both in the time and in the frequency domain.

¹This is opposed to other physical simulators that operate on individual or a few components, such as Finite Difference Time Domain, Eigenmode solvers, Eigenmode Expansion tools, Beam Propagation Methods and so on.

Second, we have investigated a novel nanophotonic reservoir computing architecture based on Photonic Crystal Cavities (PhCC). First, we have done theoretical study on these resonators (chapter 3), and proved, by comparing them to 2D FDTD simulations, that these cavities can be modeled with good accuracy using the (Temporal) Coupled Mode Theory (T)CMT. The self-pulsation that occurs in a series of two cavities with the nonlinear Kerr effect, is very similar for both modeling approaches. This means that, within reasonable assumptions (such as a small input signal bandwidth compared to the 3 dB bandwidth of the resonance), we can simulate large networks using these CMT equations without much loss of accuracy. A CMT simulation, however, takes only a few milliseconds, compared to a 10-hour FDTD simulation for the same system.

Based on these models, we have shown theoretically that passive nanophotonic reservoirs based on passive resonators are useful systems of computation. To prove this we have performed experiments using two different tasks.

In chapter 5 we have solved a speech recognition task. We have compared the PhCC architecture with an architecture based on Semiconductor Optical Amplifiers (SOAs) that was recently proposed by K. Vandoorne [1], and found that the performance is similar to the SOA reservoir (with a word error rate of 5%), which is better than solving the task using a state-of-the-art classical hyperbolic tangent reservoir. The advantage, as opposed to the SOA reservoir, is that we do not need to amplify the signal on-chip, so these reservoirs are less power consuming. This regime depends on many parameters, such as the detuning (the degree of off-resonance), the input power (compared to the characteristic nonlinear power), the resonance wavelength, cavity lifetime, the phase difference between the individual resonators, the loss in the connections and the delay between the individual resonators. The main conclusion is that cavities with slow dynamics compared to the fastest input time scale, are more phase sensitive than fast cavities, but less sensitive to the interconnection delay. Fast cavities on the other hand, reach an optimal performance when the interconnection delay is approximately equal to half the word duration of a typical speech sample. This is in correspondence with previous work by K. Vandoorne [1]. The interesting dynamics which we encountered in chapter 3, such as the self-pulsation and chaos, are unwanted. For this reason, the input power should be low compared to the nonlinear characteristic power P_0 of the reservoir.

In chapter 6, we have shown that we can train the same system in order to generate periodic patterns. The performance of the reservoir is better than the performance of a classical hyperbolic tangent reservoir: the normalized root mean square error between the target signal and the generated signal is 0.030 ± 0.021 , compared to 0.127 ± 0.111 for the case of a discrete-time real-valued reservoir in the same setup. Again, there was a huge parameter space

that had to be investigated. The main conclusion was that reducing the interaction between the cavities improved the results. This occurs when the regions of self-pulsation and chaos are avoided, i.e., when choosing a minimal delay between the cavities, adding random phases, or adding some attenuation to the connections. The total information processing capacity of this system shows the same trend: for high powers and negative detunings, the total information processing capacity of the system decreases rapidly.

7.2 Perspectives and future work

Nanophotonic RC is still in its infancy. Although it promises to be faster and more power efficient than classical RC, a lot of research on different subdomains still remains to be performed.

7.2.1 Alternative training methods

The most important work that still has to be performed is an extensive study and evaluation of the different training methods for FORCE learning. Until now, we have used the RLS rule which converges fast, but involves complex matrix manipulations, which have to be fed to a PC and have to be computed during training. As we explained in the previous chapter, other learning techniques can simplify the calculations, such as a more simple delta-type rule, or Hebbian learning [2], where the exact error signal is not explicitly needed. If these methods are successful, an experimental setup will be easier to realize.

7.2.2 New benchmark tasks and applications

In chapter 6 we have trained a nanophotonic reservoir to generate periodic patterns, following the learning methods that were introduced in [3]. We have observed very similar properties for the nanophotonic reservoir in terms of stability and convergence as for the networks that were used in the original paper (leaky hyperbolic tangent reservoirs). This suggests that the nanophotonic reservoir will also perform well on other, more complex tasks (such as those presented in [3]) which are relevant to photonics. The first one is an N-bit optical memory which can remember its state through time. In [3], a 4-bit memory is made with 8 inputs, which correspond to ON/OFF for the 4 different bits. Each output is then trained to represent one bit of the optical memory. The second application is a pattern generator controlled by static input. This can be seen as a signal encoder which can generate different patterns depending on several (analog or discrete) inputs. This signal is then put on a channel, which introduces noise, loss and nonlinearities which degrade the signal. A decoder could

be trained to compensate for the channel distortion (channel inversion), and at the same time decode the signal back to several output streams.

7.2.3 Improved modeling

Until now, we have simulated all reservoirs using a fixed carrier frequency, and we have assumed that the signal bandwidth is small compared to the characteristic wavelength-dependency of the components. For example, in ring resonators, the main influence arises from the sharp transmission peak, which also causes a sharp transition in phase. For small Q-factors of about 600 (which were typically used in this dissertation), the 3 dB bandwidth is over 300 GHz, and we can safely neglect it for slow signals. However, when using higher Q-factors, and faster input signals, this effect becomes important to investigate. In order to do this, one has to improve the used models. This can be done by adding the dispersion effects of the different components to the used models. We continue working on a carrier frequency, but instead of having a fixed scatter matrix, the response of each node is now also dependent on previous timesteps. The impulse response, which we get from inverse Fourier transforming the scatter matrix $S(p_1, p_2, \omega)$ for the frequency range we are interested in, can be convolved with the input signals. Here we have used the carrier frequency to center $S(p_1, p_2, \omega)$ around the origin before actually performing the inverse Fourier transform. In this way, we can effectively take into account wavelength-dependent effects. This will however have a large influence on the simulation time, because for each timestep, for each component, a convolution of the input signal has to be calculated. It might be useful to parallelize this operation, either by using multiple CPU cores or using GPGPU (a graphical card).

7.2.4 Measurements and experimental setup

In section 3.3 we have demonstrated that it is possible to create 1D wire photonic crystal cavities with Q-factors that are sufficient for nanophotonic reservoir computing. However, in order to create a full interconnected network, there are still some steps to perform:

- Designing two identical devices that are close together to check the influence of process non-uniformity. We have to investigate what are the main influences for resonance shifts. The target is to get these shifts below approximately 0.4 nm (see chapter 6), depending on the Q-factor of the cavities (low Q-factor means less tolerance, but less nonlinear dynamics and faster signals).
- Designing for the standard 200 nm wafers, and see if the same performance can be achieved (Q-factor, insertion loss, process uniformity).

- Characterizing the temperature-coefficient by doing experiments with different input powers and adding the temperature coefficient to the CMT model from section 3.2.1.
- Optimizing the insertion loss such that the signals can propagate further through the network.

For the experimental setup, we can use a fiber array to insert multiple signals on-chip. However, this fiber array is limited in the number of channels (8). Especially for the speech recognition task, where 77 input channels are used, this is a problem. There are other tasks, where in principle only a few inputs are needed, such as the signal generation task. Here, the main problem is how to read out the states of the reservoir in order to perform the training. Using the relatively simple Hebbian learning rule it is possible to use only one global error signal to train the reservoir, but because they have to be scaled by the actual neuron state, the issue still remains on how to read them out in order to modify the weights.

The weight modifications can be done using optical modulators. For example, using a Mach-Zehnder Interferometer or ring resonator modulator. The electrical signal modifies the refractive index of the material (in case of the MZI, this is only modified in one of the arms), which causes a change in transmission. This has been done previously, but only for a few electrical contacts. Moving towards 50-100 of these electrical contacts will be very challenging.

References

- [1] Kristof Vandoorne. *Photonic Reservoir Computing with a Network of Coupled Semiconductor Optical Amplifiers*. PhD thesis, Ghent University, 2011-2012.
- [2] Robert Legenstein, Steven M Chase, Andrew B Schwartz, and Wolfgang Maass. *A reward-modulated hebbian learning rule can explain experimentally observed network reorganization in a brain control task*. *Journal of Neuroscience*, 30:8400–8410, 2010.
- [3] David Sussillo and L. F. Abbott. *Generating Coherent Patterns of Activity from Chaotic Neural Networks*. *Neuron*, 63(4):544–557, AUG 27 2009.



Derivation of the efficient computation of the NRMSE

The calculation of the performance of the MSO task for chapter 6 is done by calculating the NRMSE for different positions of the test signal. This is computationally very intensive, but can be optimized by rewriting the equations until we have a convolution (and then calculating the convolution once, which is faster). First we define the NRMSE:

$$\text{NRMSE} = \sqrt{\frac{\frac{1}{K} \sum_{k=1}^K (z[k] - s[k])^2}{\frac{1}{K-1} \sum_{k=1}^K (s[k] - \bar{s})^2}} \quad (\text{A.1})$$

To find the convolution, we start by rewriting the MSE:

$$\begin{aligned} \text{MSE}[n] &= \langle (z_{test}[k] - s[k+n])^2 \rangle_k \\ \text{MSE}[n] &= \langle z_{test}[k]^2 \rangle_k - 2 \langle z_{test}[k]s[k+n] \rangle_k + \langle s[k+n]^2 \rangle_k \\ \text{MSE}[n] &= A - 2B[n] + C[n] \end{aligned}$$

In which A is the energy of the output signal and $C[n]$ is the energy of the tar-

get signal over a window $[n, n + 100T_1]$. As the target signal is periodic and this window is large enough, we assume $C[n]$ is constant and only needs to be calculated once. $B[n]$ can further be written as

$$\begin{aligned}
 B[n] &= \frac{1}{K} \sum_{k=0}^K z_{test}[k]s[k+n] \\
 B[n] &= \frac{1}{K} \sum_{k=-K}^0 z_{test}[-k]s[n-k] \\
 B[n] &= \frac{1}{K} \sum_{k=-K}^0 z'_{test}[k]s[n-k] \\
 B[n] &= \frac{1}{K} (z'_{test} * s)[n]
 \end{aligned} \tag{A.2}$$

Where $z'_{test}[k] = z_{test}[-k]$. This convolution can be calculated efficiently. The NRMSE is now calculated from the MSE: $NRMSE[n] = \sqrt{MSE(n)/var(s)}$, where $var(s)$ is the variance of $s[k]$. Finally, we slide the test signal over the interval $n \in [415T_1, 1415T_1]$ and select the minimal value for the NRMSE. z_{test} is the output signal when the system is in freerun for a long time. This means that if the system is not stable in long-term, the NRMSE will return a very large value. Hence, we both test for stability and quality of the output signal through the NRMSE.