A New Method to Determine Maximum Perturbation Growth in a Quasi-Geostrophic Ocean Model

Aleid Oosterwijk

July 13, 2016

Abstract

Analysing growth of initial perturbations in dynamical systems is an important aspect of predictability theory because it tells us which perturbations have the strongest influence on the system. For linear systems, these perturbation are the modes with the largest eigenvalues. For nonlinear systems, we consider the conditional nonlinear perturbation (CNOP). These can be found by solving a nonlinear constrained maximization problem, which is typically done using sequential quadratic programming (SQP), a routine that requires an adjoint model. Such adjoint models are not always available. Therefore, we study two adjoint-free methods: PSO and COBYLA. Because such methods typically work best on low-dimensional problems, we apply dimension reduction. We use the proposed methodology to find the CNOPs of a quasi-geostrophic ocean model. We find that COBYLA outperforms PSO and is able to find reasonable CNOPs, although at a higher computational cost than conventional adjoint-based methods.

1 Introduction

Dynamical systems are widely used to describe physical systems such as the atmosphere, ocean and biological processes. The evolution and future states of the system depend on the initial state: variations in the initial state can lead to different future states. Studying the growth of initial perturbations is important in predictability theory, for example in weather forecasting. The European Centre for Medium-Range Weather Forecast (ECMWF) uses linear singular vectors (LSV) for generating the fastest growing initial perturbations for the ensemble forecast [13]. The LSV method is based on a linearisation of the system, the tangent linear model (TLM) [5]. The validity of this linearisation however, is subject to discussion [13]. Therefore, nonlinear equivalents of the LSV have been developed to determine the fastest growing initial perturbations, for example nonlinear singular values (NSVA) [12], and conditional nonlinear optimal perturbations (CNOPs) [13]. The CNOP can be obtained by maximising an objective function subject to constraints. This is a constrained optimisation problem and can be solved by several numerical optimisation techniques. A common method is sequential quadratic programming (SQP) [13, 14, 18]. This method however, is only suitable for models for which an adjoint model is available. The adjoint model is the transpose of the tangent linear model, and can be used to obtain gradients of the objective function with respect to the initial condition, from gradients with respect to the output [8]. In this way the initial input which optimises the objective function can be efficiently determined. However, for many models the calculation of the adjoint model is not straightforward. For these models derivative-free methods are needed.

In this paper two of such methods are investigated, particle swarm optimisation (PSO) and constrained optimisation by linear approximation (COBYLA). PSO has recently been successfully applied for determining CNOPs [11]. PSO is inspired on the behaviour of swarms, such as flocks of birds or schools of fish, in which each individual seeks to improve its

position and responds to the behaviour of its neighbours. COBYLA [16] is based on the Nelder-Mead simplex method [14], but adapted for constrained optimisation. Also other derivative-free optimisation methods could be considered to find CNOPs of models for which no adjoint model is available. One could think of adapted local search methods, such as simulated annealing, or other ensemble based methods, such as genetic algorithms. Many of these are described in Nocedal and Wright [14]. The efficiency of these methods largely depends on the size of the search space, i.e. the dimension of the domain of the objective function. The performance of derivative-free optimisation algorithms quickly decreases when the dimension of the search space is increased. This is known as 'the curse of dimensionality', which is less present in derivative-based methods. Therefore, in order to apply PSO and COBYLA to large systems, dimension reduction is necessary. Based on Mu et al. [11], this will be done applying PCA (principal component analysis).

As shown by Mu et al. [11], a combination of PCA and PSO can be applied to calculate CNOPs of the ZC model, to find the optimal precursor of ENSO events. They show that an increase of the search space dimensionality leads to a smaller error and larger running time. The optimal amount of dimensions is found to be 30, the original dimension of the search space is 1080. The spatial structure of the CNOPs found is comparable to the CNOPs found by SQP, the objective value is significantly lower due to the dimension reduction, but converges to an acceptable solution in 85% of the cases. The runtime is approximately three times larger for the PCA based PSO than for SQP.

The derivative-free methods under investigation, COBYLA and PSO, are compared, in terms of accuracy and speed, to SQP, which requires the adjoint model. The methods are applied to a quasi-geostrophic ocean model. In Section 2 the model and the CNOP will be defined. In Section 3 we will explain the algorithms of PSO and constrained optimization by linear approximation (COBYLA) [16], as well as the dimension reduction methods. In Section 4 the effect of different termination conditions and dimensionality on the performance of COBYLA is discussed, as well as the results for PSO. Then we will conclude that COBYLA is able to obtain the CNOPs, albeit at higher computational cost and with less accuracy than SQP. We will end with a discussion about the performance of COBYLA and the applicability of COBYLA to solving CNOPs in other systems.

2 Theory

The explanation of CNOP is based on Mu et al. [13] and Terwisscha van Scheltinga and Dijkstra [18].

2.1 CNOP

Assume the system can be described by the following model, which is discretised in space:

$$\frac{\partial \boldsymbol{w}}{\partial t} + F(\boldsymbol{w}) = 0, \tag{1a}$$

$$\boldsymbol{w}|_{t=0} = \boldsymbol{w}_0 \tag{1b}$$

where $\boldsymbol{w}(t) = (w_1(t), \ldots, w_n(t))$, with $(\boldsymbol{w}_i, t) \in \Lambda \times [0, T]$, F is a nonlinear operator, $\boldsymbol{w}_0$ is the initial state, Λ is a domain in $\mathbb{R}^d$ and $T \in \mathbb{R}^+$. Note that in our implementation $\boldsymbol{w}$ is the state vector consisting of values of ψ at each gridpoint and $d = 2400$. Suppose the initial value problem is well-posed and $\mathcal{M}$ is the nonlinear propagator from 0 to time T, so $\boldsymbol{w}(T) = \mathcal{M}(\boldsymbol{w}_0)(T)$ is well-defined. Let $\bar{\boldsymbol{x}}(t)$ and $\tilde{\boldsymbol{x}}(t)$ be two solutions of the system, with initial conditions $\bar{\boldsymbol{x}}_0$ and $\bar{\boldsymbol{x}}_0 + \boldsymbol{x}_0$, respectively. Choose $\bar{\boldsymbol{x}}_0$ a specific (in our case, steady state) solution to the system and $\boldsymbol{x}_0$ a small initial perturbation to this state. Integrating the state and the perturbed state is done by applying the nonlinear propagator, i.e. $\mathcal{M}(\bar{\boldsymbol{x}}_0)(T)$ and $\mathcal{M}(\bar{\boldsymbol{x}}_0 + \boldsymbol{x}_0)(T)$.

The goal is to find the initial perturbation which causes the largest perturbation after time T. Of course, the initial perturbation should be restricted to study physically realistic situations. We choose the constraint condition $||\boldsymbol{x}_0|| \leq \delta$, for a chosen norm $||.||$. The conditional nonlinear optimal perturbation (CNOP) is defined as the perturbation $\boldsymbol{x}_0^\delta$ such that

$$J(\boldsymbol{x}_0^\delta) = \max_{||\boldsymbol{x}_0|| \leq \delta} J(\boldsymbol{x}_0), \tag{2}$$

where

$$J(\boldsymbol{x}_0) = ||\mathcal{M}(\bar{\boldsymbol{x}}_0 + \boldsymbol{x}_0)(T) - \mathcal{M}(\bar{\boldsymbol{x}}_0)(T)||. \qquad (3)$$

We see that finding the CNOPs of a system is a constrained optimisation problem.

2.2 Quasi-Geostrophic Model

We will apply our techniques to a quasi-geostrophic ocean model, as described in detail in Terwisscha van Scheltinga and Dijkstra [18].

The model has a domain which is a rectangular ocean basin of size $L \times L$ and depth D. It is situated on a midlatitude β-plane with $\theta_0 = 45°N$, $f_0 = 2\Omega \sin\theta_0$ where Ω is the angular velocity of the rotation of the Earth. The meridional gradient of the Coriolis parameter is denoted by β_0. We assume constant density and a wind-forcing $\boldsymbol{\tau} = \tau_0[\tau^x(x,y), \tau^y(x,y)]$, where τ_0 is a typical amplitude. The symmetric wind-stress profile is given by

$$\begin{aligned} \tau^x &= -\cos(2\pi y/L), \\ \tau^y &= 0. \end{aligned}$$

The governing equations are made dimensionless. This is done with horizontal length scale L, vertical length scale D, horizontal velocity scale U, advective timescale L/U and characteristic wind stress vector amplitude τ_0.

The dimensionless barotropic quasi-geostrophic model of the flow for the vorticity ζ and the geostrophic streamfunction ψ is

$$\begin{aligned} &\left[\frac{\partial}{\partial t} + u\frac{\partial}{\partial x} + v\frac{\partial}{\partial y}\right][\zeta + \beta y] \\ &\qquad = Re^{-1}\nabla^2\zeta + \alpha_\tau\left(\frac{\partial \tau^y}{\partial x} - \frac{\partial \tau^x}{\partial y}\right) \qquad (4a) \\ &\zeta = \nabla^2\psi, \qquad (4b) \end{aligned}$$

which can be found in Dijkstra [4] as (6.3), where we use that there is no bottom topography, no bottom friction and a rigid lid approximation, which means we assume no surface deformation.

The horizontal velocities are given by

$$u = -\frac{\partial \psi}{\partial y}, \qquad v = \frac{\partial \psi}{\partial x}.$$

The parameters in the governing equation, Re, the Reynolds number, β, the planetary vorticity gradient parameter and α_τ, the wind stress forcing strength, are defined as follows.

$$Re = \frac{UL}{A_H}, \qquad \beta = \frac{\beta_0 L^2}{U}, \qquad \alpha_\tau = \frac{\tau_0 L}{\rho D U^2}.$$

Here g is the gravitational acceleration, A_H is the lateral friction coefficient. Under the assumption of a Sverdrup balance, which means the vorticity change by wind stress is compensated with meridional transport, the horizontal velocity scaling becomes $U = \frac{\tau_0}{\rho D \beta_0 L}$, from which it follows that $\alpha_\tau = \beta$. All parameter values are chosen equal to the values in Terwisscha van Scheltinga and Dijkstra [18].

The boundary conditions are given by

$$\begin{aligned} \psi = \frac{\partial \psi}{\partial x} = 0 \qquad &\text{at } x = 0 \text{ and } x = 1, \qquad (5a) \\ \psi = \zeta = 0 \qquad &\text{at } y = 0 \text{ and } y = 1. \qquad (5b) \end{aligned}$$

The space-discretisation is done using second order central differences. The gridsize that is used in the implementation is 60×40. This implies that our state vector, which consists of the ψ values at the gridpoints, has length 2400. Time integration is done using Crank-Nicolson [18], with time-steps of 0.001, or in dimensional units, 1.75 days. The CNOPs will be calculated for $Re = 25$ and $Re = 50$. The steady states of the model at these values of Re are shown in Figures 1 and 2.

The norm that is implemented is the kinetic energy norm as used in Terwisscha van Scheltinga and Dijkstra [18], which is based on the kinetic energy $E = \frac{1}{2}\int_V (\boldsymbol{u}_0^2 + \boldsymbol{v}_0^2)\mathrm{d}x\mathrm{d}y$.

For the discretised QG model with state vector $\boldsymbol{x}_0$ the kinetic energy norm is calculated as follows. Let $\boldsymbol{K}$ be a linear operator that maps $\boldsymbol{x}_0$ to the velocity vector, calculated from neighbouring gridpoints. The

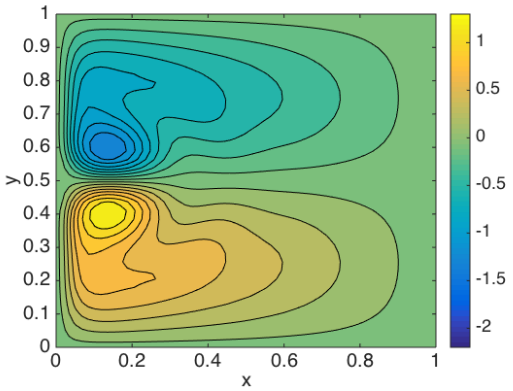


Figure 1: Streamfunction of the steady state at $Re = 25$ of the quasi-geostrophic ocean model

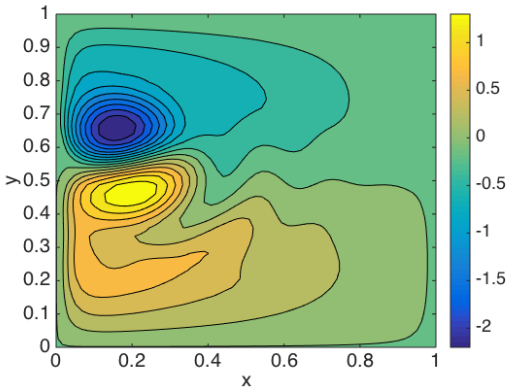


Figure 2: Streamfunction of the jet-up steady state at $Re = 50$ of the quasi-geostrophic ocean model

kinetic energy norm[1] then is

$$\begin{aligned} ||\boldsymbol{x}_0||_E =& \frac{1}{2}\Delta x \Delta y (\boldsymbol{K}\boldsymbol{x}_0 \cdot \boldsymbol{K}\boldsymbol{x}_0) \\ =& \frac{1}{2}\Delta x \Delta y ||\boldsymbol{K}\boldsymbol{x}_0||_2^2. \end{aligned} \tag{6}$$

3 Methods

We will compare the performance of the adjoint-free methods PSO (which has been implemented in Mu et al. [11] the calculate CNOPs) and COBYLA [16] to SQP, an algorithm that requires the adjoint model. A detailed description of the SQP method can be found in Mu et al. [13], Nocedal and Wright [14]. An application of the SQP method to the QG model can be found in Terwisscha van Scheltinga and Dijkstra [18]. The algorithms of PSO and COBYLA are described below, and are illustrated using a 2D optimisation problem.

3.1 Particle Swarm Optimisation

Following the methods in Mu et al. [11] we have implemented PSO to solve the optimisation problem. PSO is based on swarm behaviour. The algorithm works as follows. Start with a randomly chosen, uniformly distributed, group of points in the domain, called the swarm. Each point has a velocity, chosen randomly from a uniform distribution. In each iteration, the objective value of each particle is evaluated. If there is improvement in objective value, the best global position and the best position of each particle is updated. The particle speeds are adapted to go towards the best global and the best local position. Subsequently the particle positions are updated using the newly determined velocities, checking the constraint condition and, if necessary, adapting positions to meet the constraint. Then the objective values are calculated again, starting the next iteration. An illustration of the PSO algorithm in 2D is shown in Figure 3. A more detailed description of the implementation can be found in Nocedal and Wright [14]. The advantage of this method is that it is very

[1] Note that this is actually a 2-seminorm. However, it can still be used to measure the constraint and objective value.

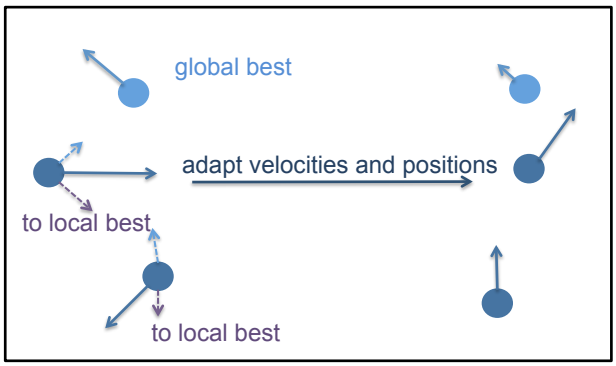


Figure 3: Illustration of PSO algorithm iteration in 2D. The blue dots denote particles. The darker blue arrows denote their velocities. The blue dashed arrows denote the velocity towards the global best position. The purple dashed arrows denote the velocity towards the local best position. The lighter blue dot denotes the particle at the global best position.

intuitive and requires no further information on the problem other than the constraint and objective function. The drawback is that it might not be optimal in terms of accuracy and speed, the algorithm is very sensitive to the parameter choices. Some literature is available determining some of the optimal parameter settings. In this study we used the values described in Li-Ping et al. [10]. However, many parameters are problem-dependent and are therefore unknown a priori. A few workarounds have been added to make sure the algorithm does not get trapped in a local optimum prematurely, but still there is no guarantee that the solution found is indeed the global optimum. An example of a workaround is the inclusion of 'scouts', based on the cognitive-only version in Engelbrecht [6] and different ways of handling the constraint, as described in Coath and Halgamuge [3].

3.2 COBYLA

Constrained optimization by linear approximation, COBYLA, is an optimisation algorithm that iteratively defines a simplex on which the objective function is linearised and optimised, after which a new simplex is chosen. The new simplex is chosen in order to improve the objective value or the shape of the simplex. The size of the simplex vertices is reduced during the process, starting from size ρ_{start}. The process terminates when the vertices have size ρ_{end}. An illustration of this process in 2D is shown in Figure 4. The constraint is implemented using a penalty function, which is adapted during the iteration process. A detailed description of the method can be found in Powell [16]. COBYLA is based on Nelder-Mead [14]. For the Nelder-Mead method convergence has been proved for strictly convex functions in one and two dimensions [9]. This clearly does not apply to our objective function, but Nelder-Mead is widely used and generally shows good convergence results [14]. The COBYLA routine has been implemented using $\rho_{start} = 0.1$, and varying ρ_{end}. The method is initialised with a position in the search space, randomly chosen from a uniform distribution. A step-by-step guide for the implementation of COBYLA in combination with PCA can be found in Appendix B. Comparing COBYLA to PSO, observe that in COBYLA there are much less parameters that can be tuned. Only the initial step size and the final step size can be chosen, ρ_{start} and ρ_{end}. This step size translates roughly to the size of the simplex. Another difference with PSO is that while both algorithms find only local optima, PSO can be thought of as more global. This is because PSO covers a large parts of the search space simultaneously. This means that theoretically we would expect PSO to perform better on functions with many local optima, while we expect COBYLA needs less objective function evaluations. As a last remark, it should be noted that we do not expect PSO and COBYLA to perform as good as the SQP method, as this makes use of the adjoint model, while PSO and COBYLA do not.

3.3 Results 2D Optimisation Problem

To illustrate COBYLA and PSO, a suitable 2D constraint optimisation problem has been selected. It is based on the Simionescu function [17], which is adapted to be slightly asymmetric. It is a minimisa-

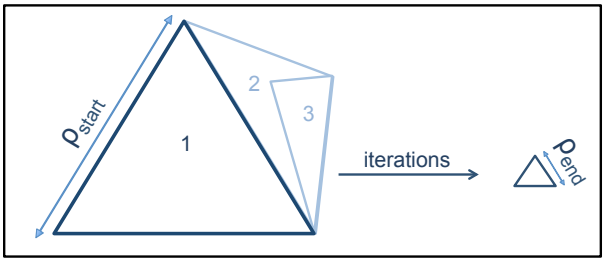


Figure 4: Illustration of COBYLA algorithm in 2D. The initial simplex with vertices of size ρ_{start} is denoted by 1, subsequent simplexes in light blue are denoted by 2 and 3. The final simplex has vertex length ρ_{end}.

tion of

$$F(x, y) = 0.1xy,$$

subject to the constraint

$$x^2 + y^2 \leq \left(1 + 0.2\cos\left(8\arctan\left(\frac{x}{y + 0.1}\right)\right)\right)^2.$$

In Figures 5 and 6 an example run of both PSO and COBYLA is shown on this test function. The function values of the Simionescu function are depicted by the colors, the white shape is the constraint boundary and the plus-signs denote objective function calls. In Figure 5 all particles are shown, their colors changing from dark blue in the first iteration to dark red in the last iteration. In Figure 6 one optimisation run by COBYLA is shown, the final solution shown as the red plus-sign. Note that PSO uses many more objective function calls than COBYLA. This implies that the algorithm is less efficient. On the other hand, we see that PSO finds both the optimal and the sub-optimal solution, while COBYLA finds only the optimal one. This means that while COBYLA could get stuck in the sub-optimal solution, PSO finds both and therefore will be more likely to return the right optimum. Furthermore, note that if one would be interested in finding multiple good solutions, instead of only the global optimum, PSO would therefore be more efficient as it finds several good solutions in one

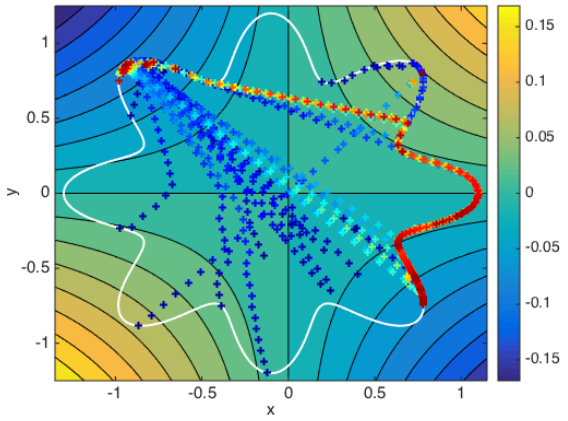


Figure 5: Example PSO optimisation run on Simiunescu function with contour plot of function values and constraint boundary in white. The plus-signs denote objective function calls. All particles are shown, depicted in dark blue in the first iteration, changing color up to red in the last iteration.

optimisation run. Of course this result could also be obtained by running COBYLA several times with different initial conditions.

3.4 Dimension Reduction

Because in general derivative-free optimisation algorithms do not perform very well on large-dimensional search spaces, a dimension reduction is necessary before applying COBYLA and PSO. As argued in Osborne and Pastorello [15], in a driven dissipative system like the QG model, any state can be approximated in a low dimensional space. This space consists of attractors of the system. The system can, after long evolution, get into a steady state that consists of these attractors. Using a training set, a matrix consisting of data of a long evolution, we obtain these attractors as the principal components of the data set. According to Osborne and Pastorello [15], these attractors are a good low-dimensional approximation to the system. The original search space has 2400 dimensions (40×60). This we will reduce to 50 up to 200 dimensions, to see which amount of dimensions yields the optimal results during optimisation.

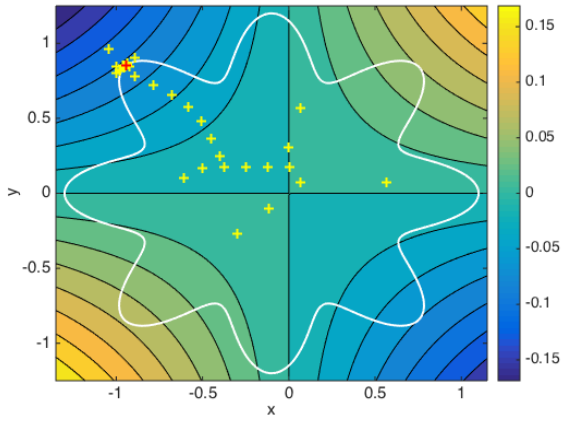


Figure 6: Example COBYLA optimisation run on Simiunescu function with contour plot of function values and constraint boundary in white. The plus-signs denote objective function calls. The red plus sign is the final solution.

PCA is implemented as dimension reduction method, because it is a widely used and effective dimension reduction method. PCA in combination with PSO has been implemented for a different model in Mu et al. [11]. For PCA, a training set is used, which is a matrix of a data set generated by integrating the model for 10 years with an additional noise on the wind-stress. The dimensionless wind stress is $\tau^x = -\cos(2\pi y) + r_t \zeta \sin(2\pi y)$ where $r_t = 0.1$ is the wind stress parameter and ζ is a random variable from a Gaussian distribution with mean 0 and variance 1. In this way the training set is obtained. Each column of the training set is a state vector at a certain time. All rows of the training set are centered (the mean of each grid point is shifted to zero), to obtain the centered training set $\boldsymbol{X}$. The principal components (PCs) are obtained using an eigenvalue decomposition of the matrix $\boldsymbol{X}\boldsymbol{X}^T$, which is similar to the covariance matrix of $\boldsymbol{X}$.

$$\boldsymbol{X}\boldsymbol{X}^T\boldsymbol{a} = \lambda\boldsymbol{a}$$

The eigenvectors $\boldsymbol{a}_i$, $i = 1, \ldots, d$ obtained this way are called empirical orthogonal funcions (EOFs). The EOFs corresponding to the largest k eigenvalues account for a fraction of the variance in the data equal to the fraction of the sum of the corresponding eigenvalues and the sum of all eigenvalues. So PCA is a reduction method that conserves as much variance as possible. The principal component $\boldsymbol{c}_i$ corresponding to the EOF $\boldsymbol{a}_i$ is the projection of $\boldsymbol{X}$ onto this EOF:

$$\boldsymbol{c}_i = \boldsymbol{X}\boldsymbol{a}_i$$

Now any state $\boldsymbol{x}$ can be approximated in k dimensions as

$$\boldsymbol{x} \approx \boldsymbol{x}_R = \sum_{i=1}^{k} \alpha_i \boldsymbol{a}_i$$

where $\boldsymbol{x}_R$ is the reduced state, $\boldsymbol{a}_i$ is the ith EOF and α_i is the contribution of $\boldsymbol{a}_i$ to $\boldsymbol{x}_{red}$. When $k = d$, the coordinate transformation is exact and $\boldsymbol{x}_{red} = \boldsymbol{x}$. A detailed description of PCA can be found in Hannachi [7].

To quantify the loss of precision due to dimension reduction we introduce a measure of the relative error like in Mu et al. [11]:

$$E_R = \frac{||\boldsymbol{X} - \boldsymbol{X}_R||_2^2}{||\boldsymbol{X}||_2^2} \tag{7}$$

where $\boldsymbol{X}_R$ is the approximation of $\boldsymbol{X}$ in reduced dimensionality, $\boldsymbol{X}_R = \sum_{i=1}^{k} \boldsymbol{X}\boldsymbol{a}_i$.

The dimension reduction is done for a training set for the case $Re = 25$ and a training set for the case $Re = 50$. These are the regimes for which the CNOPs will be calculated. The CNOPs as calculated with SQP are projected onto the EOFs and the contribution of the principal components to the CNOPs are shown in Figure 7. Both for $Re = 25$ and for $Re = 50$ we observe a steep decrease in contribution in approximately the first 50 PCs, showing that at 100 PCs already the contribution is quite low. The contribution of PCs between 200 and 400 is larger for $Re = 50$ than for $Re = 25$. From 500 PCs onward the contribution seems comparable again. However, we are not able to draw strong conclusions about the differences in the projections for both Reynolds numbers. It should be noted that the variance of the training sets is small. Only as many as 10 PCs are needed to account for more than 90% of the total variance in the data sets, for both $Re = 25$ and $Re = 50$.

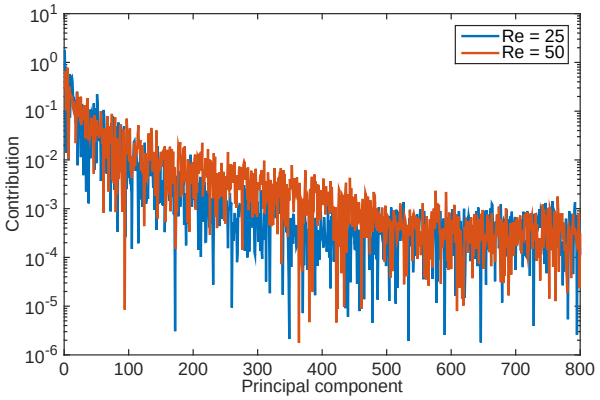


Figure 7: Contribution of first 800 principle components to the CNOP calculated by SQP as shown in Figures 10 and 9, projected onto the EOFs for $Re = 25$ and $Re = 50$, respectively.

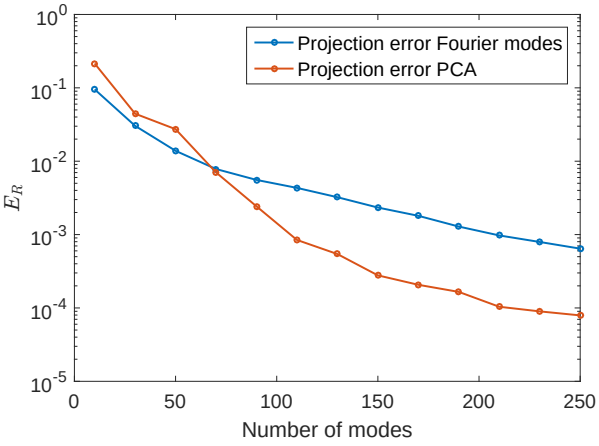


Figure 8: Error of projected CNOP against number of Fourier modes and principal components

3.4.1 Fourier modes

In addition to PCA, also other methods to reduce the dimensionality of the problem have been considered. Fourier modes have also been used as basis functions instead of EOFs. These are defined as

$$\phi_{nm} = \sin(n\pi x)\sin(m\pi y), \tag{8}$$

with $x, y \in [0, 1]$ and $n, m \in \mathbb{N}$. The effectiveness of Fourier modes has been compared to principal component analysis by reducing a CNOP found by the SQP algorithm to a varying number of dimensions. The Fourier modes are ordered by increasing k, $k = n^2 + m^2$, so that $\phi_{22} < \phi_{31}$. Note that the use of Fourier modes is restricted by the gridsize. When, for example, using $m = 40$, aliasing will occur as the y-axis is discretised in 40 steps.

In Figure 8 we see that up to 80 dimensions, the reduction method using Fourier modes leads to a smaller error. However, when the projected space is larger than 80 dimensions, it is favourable to use PCA. This means that PCA will be used in this study, as this would for more precise results be the best reduction method. Another advantage of PCA is that it would work for any kind of data, while Fourier modes only work because the streamfunction field can be easily written as sums of Fourier modes. An advantage of Fourier modes is that no pre-computation is necessary, contrary to PCA.

4 Results

In this section we will compare the performance of the adjoint-free methods COBYLA and PSO to the adjoint-based method SQP, which is used as a benchmark. The CNOPs that are calculated with all methods for the comparison are the CNOP at $\delta = 0.1$, $T = 7$ days, $Re = 25$, and the CNOP at $\delta = 0.1$, $T = 7$ days, $Re = 50$. Different values of Re are taken into account, to study different stability regimes. In this way the influence of the stability of the model on the performance of the optimisation methods can be observed. The CNOPs for these settings, as calculated by SQP, are shown in Figures 9 and 10. The SQP algorithm needs approximately 600 objective function calls to find the CNOP for $Re = 25$, and 200 objective function calls to find the CNOP for $Re = 50$. The solution of the SQP algorithm is reliable, as the algorithm performs very consistently. In Terwisscha van Scheltinga and Dijkstra [18] the solution is assumed to be the correct one, i.e. the global maximum, in this study this assumption is done as well. In this way we can assess the

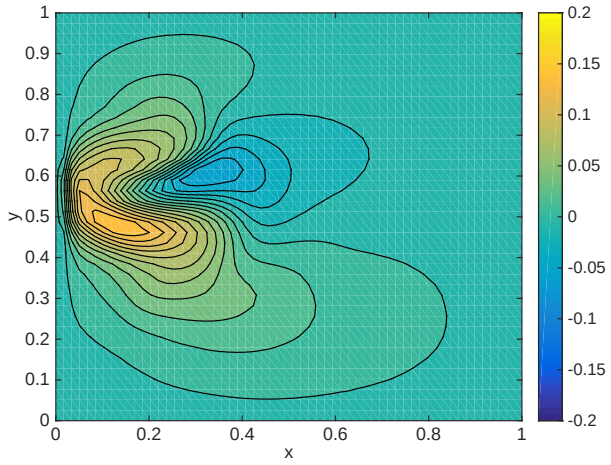


Figure 9: CNOP of jet-up state at $Re = 50$ with $\delta = 0.1$ and $T = 7$ days. The colors depict the streamfunction values at the gridpoints. x, y are the normalised spatial coordinates.

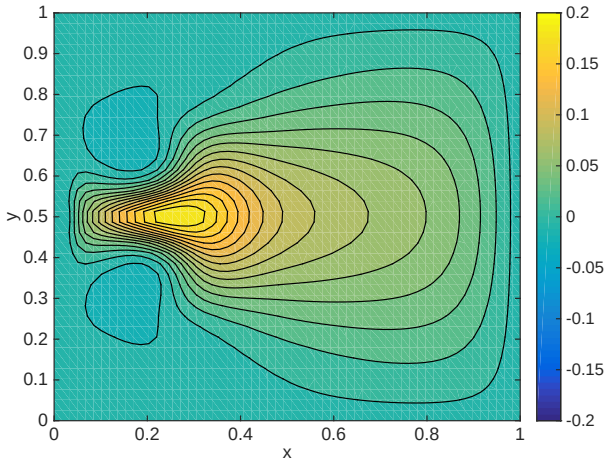


Figure 10: CNOP of at $Re = 25$ with $\delta = 0.1$ and $T = 7$days. The colors depict the streamfunction values at the gridpoints. x, y are the normalised spatial coordinates.

solutions of COBYLA and PSO by their resemblance to the SQP solution.

Different parameter settings and dimensions have been tested in COBYLA and have been compared to SQP. First, we will discuss the different values of ρ_{end} that have been used. Secondly the different search space dimensions that have been tested for both $Re = 25$ and $Re = 50$. Before turning to the results, we will discuss how the results will be presented. All results are obtained by applying the method concerned 10 times to the optimisation problem, reporting the average and best solution. The objective value and number of objective function calls are reported, as well as the 'error norm', which is a measure for the error in the solution obtained with respect to the SQP solution for that problem. The error norm is defined as

$$\text{Error norm} = \frac{||\boldsymbol{x}_C - \boldsymbol{x}_{SQP}||_2}{||\boldsymbol{x}_{SQP}||_2},$$

where $\boldsymbol{x}_C$ is the solution found by COBYLA and $\boldsymbol{x}_{SQP}$ is the solution as found by SQP. Actually, the CNOPs come in pairs $\{\boldsymbol{x}_0^\delta, -\boldsymbol{x}_0^\delta\}$ which both have the same objective value. As COBYLA could find either of these solutions, the error norm is calculated with respect to the closest of these solutions. The results that are shown in graphs are also included as tables in Appendix A.

The error norm is included because in this case our interest lies not with the best objective value (in which case the objective value would be sufficient to measure the accuracy of the methods) but the state (initial perturbation) that yields this objective value. This means that if the objective value is nearly as good as the global maximum value, but the value is found at a local maximum far from the global maximum, the state will not resemble the CNOP at all. This means the performance of the method is low even though the objective value is nearly perfect. Therefore, the norm is included to see how close the solution is to the real CNOP.

It should be noted however, that the importance of the CNOP lies in its physical features, which means that as long as the global pattern of the CNOP resembles the true solution, the solution is satisfactory, even if the value of the error norm is relatively large. In Figure 11 the solution of COBYLA can be seen for restarted COBYLA from 30 to 100 dimensions. This is as an illustration of why the norm is not sufficient

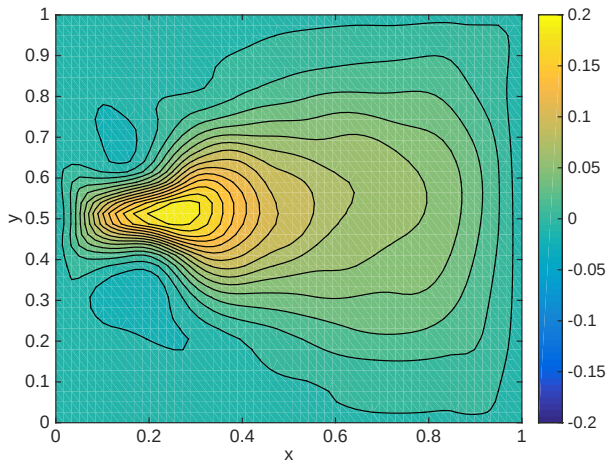


Figure 11: Example solution for $Re = 25$, $T = 7$ days and $\delta = 0.1$. This is the best solution of restarted COBYLA with 30 to 100 principal components. Its error norm is 0.3253. The colors depict the streamfunction values at the gridpoints. x, y are the normalised spatial coordinates.

to judge the results: the solution is clearly different from the true CNOP, however the resemblance can be easily recognised. Therefore also the *ratio* is calculated for all methods at $Re = 25$. This is the ratio of solutions that are, as determined by eye, similar to the real (SQP) solution. The ratio at $Re = 50$ is not determined, as this pattern is more complex and therefore it is difficult to judge the similarity.

After discussing the results for different settings of COBYLA the performance of PSO is discussed.

4.1 Different parameter settings in COBYLA

For $\rho_{start} = 0.1$ and 100 PCs, different values of ρ_{end} have been tested. The results of the optimisation are presented in Figure 12. The objective value (both average and best) increases with decreasing ρ_{end}, while the number of objective function calls increases with decreasing ρ_{end}, as might be expected. The error norm decreases with increasing ρ_{end} as expected, however its confidence bound increases and then decreases again, which is remarkable. The 10 values of which the average is taken are shown in Figure 12c, and they are observed to be in two distinct groups for $\rho_{end} = 10^{-4}$. The explanation for this is that even though the objective values of both groups are comparable, the position at which those objective values are found, differ. The two groups represent two different optima, of which one is the global optimum and the other is a local optimum. In Table 1 we see how many out of 10 runs of COBYLA for varying ρ_{end} are correct. As expected, this ratio increases as ρ_{end} decreases. Only for $\rho_{end} = 10^{-5}$ COBYLA seems reliable.

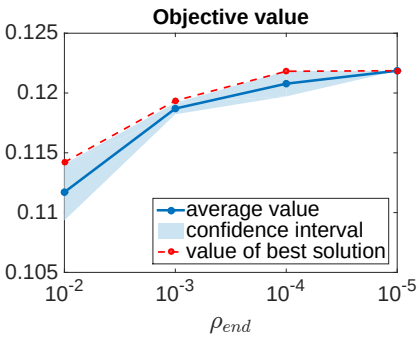
ρ_{end}	10^{-2}	10^{-3}	10^{-4}	10^{-5}
Ratio	0.0	0.2	0.5	1.0

Table 1: Ratio of correct solutions and number of total solutions of COBYLA, using 100 PCs, $Re = 25$, $\rho_{start} = 0.1$ and varying ρ_{end}.

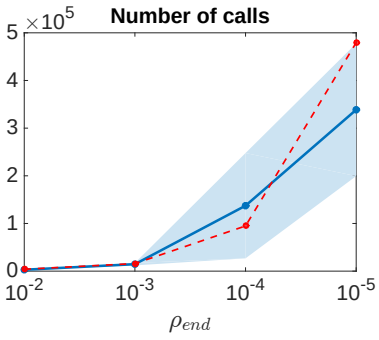
To conclude, for $\rho_{end} = 10^{-5}$ COBYLA yields the best and most reliable results, but for $\rho_{end} = 10^{-4}$ COBYLA also yields good results in half of the cases and has significantly shorter run time. It should be taken into consideration that this result might be dependent on the amount of principal components used.

4.2 Dimensionality in COBYLA

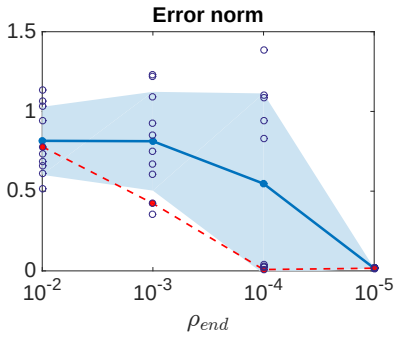
The performance of COBYLA has been tested for different amounts of principal components, for both $Re = 25$ and $Re = 50$. First the results for $Re = 25$ are discussed. Subsequently the results for $Re = 50$ are presented. The results for $Re = 25$ are shown in Figure 13, for $Re = 25$ the dimensions tested are 30, 50, 100, 150, 200 and 250. In Table 2 the ratio of correct solutions and number of total solutions is given. The minimum number of PCs needed to find the solution at the global optimum is 80. However, using 100 PCs leads to better results, and 150 PCs or more is almost a guarantee for finding the global optimum. Of course using more principal components also leads to a much more objective function calls, as more principal components



(a) Objective value



(b) Number of objective function calls made



(c) Error norm with respect to SQP solution

Figure 12: Best and average out of 10 with confidence interval of 0.68 of COBYLA, using 100 PCs, $Re = 25$, $\rho_{start} = 0.1$ and varying ρ_{end}. The dots in (c) denote the individual runs.

means more dimensions in the search space.

It is very remarkable that the solution for $Re = 30$ is significantly closer to the correct solution than the solution at $Re = 50$. This implies that the optimisation landscape is more similar to the real landscape using 30 PCs than using 50 PCs. This is a coincidence of course, as the solution in the reduced dimensional landscape should approximate the true solution better when the number of dimensions is increased. The only conclusion to draw from this result is that more than 50 PCs are needed to find the global optimum.

Another observation from Figure 13c is that for 100 principal components the confidence interval is very large. The explanation for this is given in the section discussing varying ρ_{end}. It should be noted that, even though not very well visible in Figure 13c, a slight increase in the confidence interval is visible for 250 principal components. A detailed picture of this case has been included in Figure 14. It is observed that while the best solution is still decreasing in error norm, some solutions have a larger error norm again. When inspecting the CNOPs found, it is indeed visible that the resemblance between the solutions is decreasing. This suggests that the algorithm is not converging as well in 250 dimensions as it is in 150 or 200 dimensions. An explanation for this could be that COBYLA starts suffering from the 'curse of dimensionality' at this point.

Now the results for $Re = 50$ are discussed. The results for $Re = 50$ are shown in Figure 15, for $Re = 50$ the dimensions tested are 50, 100 and 200. From Figure 15 a and b we again observe an increase in objective value and number of objective function calls as the number of principal components included increases. This is according to expectation. However, the error norm decreases only slightly. This suggests that more principal components are needed to resolve the CNOP at $Re = 50$, which is in accordance with the observation that the principal component contribution between 200 and 400 is stronger at $Re = 50$ as shown in Figure 7. However, when inspecting the solutions using 200 principal components, the patterns are very much alike the CNOP found by SQP. This suggests that even though the error norm is quite large, and there are clear differences between the solutions, the solutions found by

PC	30	50	80	100	150	200	250
Ratio	0.0	0.0	0.2	0.5	1.0	1.0	1.0

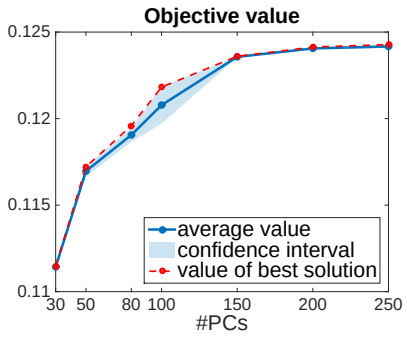
Table 2: Ratio of correct solutions and number of total solutions of COBYLA, $Re = 25$, $\rho_{start} = 0.1$, $\rho_{end} = 10^{-4}$ and varying the amount of principal components.

COBYLA might serve their purpose when we are only interested in the global pattern of the CNOP.

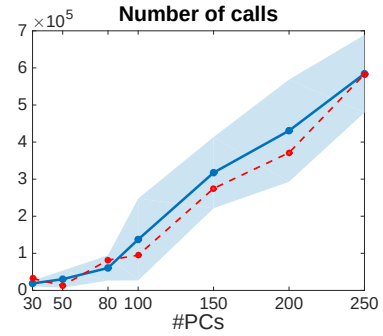
We already concluded that up to 150 principal components, some solutions are in a local optimum that is not the global optimum. The increase in principal components leads to more solutions in the right optimum. This is a qualitative effect. However, we would also like to investigate the qualitative effect on accuracy when the amount of principal components are increased. Therefore, for $Re = 25$ the convergence of only the solutions that are in the right optimum has also been investigated. The results are presented in Figure 16. It can be observed that objective function seems to be almost convergent at 250 dimensions, the number of objective function calls seems like a linear relation to the number of principal components. The error norm of the best solution is decreasing with the number of principal components. The average value is increasing at 250 principal components, as is its confidence interval. Only including the runs that converged to the global optimum, we conclude that the error norm converges to a value around 0.02 instead of zero. This can be explained partly by a difference in implementation between the methods, we will return to this in the discussion. But again, since the error norm starts diverging slightly at 250 dimensions again, we have to conclude that more dimensions will not necessarily lead to a more accurate solution. However, to draw any definitive conclusions, this should be further investigated using more principal components.

4.3 Restarted COBYLA

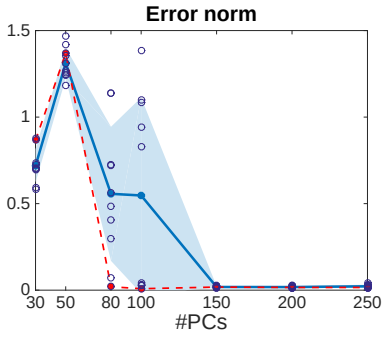
We have seen that, in general, a large dimensionality leads to more accurate solutions, while small dimensionality leads to fast convergence. This suggests that



(a) Objective value



(b) Number of objective function calls made



(c) Error norm with respect to SQP solution

Figure 13: Best and average of 10 with confidence interval of 0.68 of COBYLA, using $\rho_{start} = 0.1$, $\rho_{end} = 10^{-4}$, $Re = 25$ and varying the number of PCs. The dots in (c) denote the individual runs.

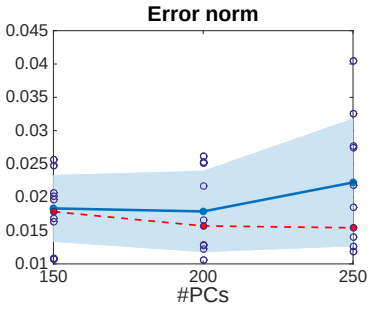
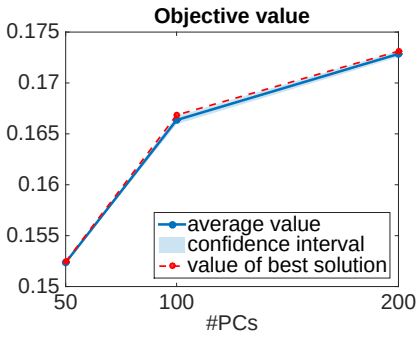
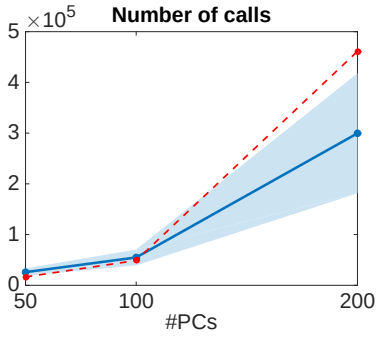


Figure 14: Best (red) and average (blue) of 10 with confidence interval (shaded blue area) of 0.68 of COBYLA, using $\rho_{start} = 0.1$, $\rho_{end} = 10^{-4}$, $Re = 25$ and varying the number of PCs, zoomed in at the section of 150 to 250 PCs. The dots denote the individual runs.

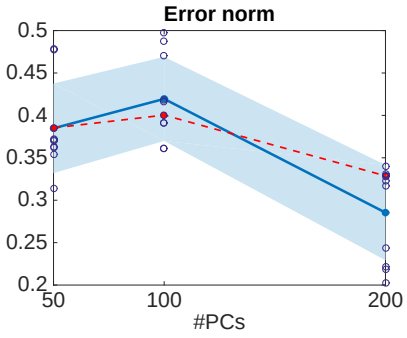
it might be efficient to first solve a low dimensional version of the problem, and restarting COBYLA in a higher dimension using the low dimensional solution as a starting point. This has been done for initial dimensionality 30, using the solution as initial point in 100 dimensions, and for initial dimensionality 50, using the solution as initial point in 100 and 200 dimensions. For all implementations, a small value of ρ_{start} of 0.01 is used, as it is to be expected that the solution is closer to the real solution than a random solution would be. The results can be seen in Figure 17. It is very remarkable, that using a solution obtained in 30 dimensions as initial point for the restart leads to good results, while using a solution obtained in 50 dimensions does not. This is however in agreement with the results in Figure 13, where we observed that the solution with 30 PCs resembled the correct solution more than the solution with 50 PCs. This means that even though the objective value of this solution is worse, it is still a better restart point for the larger dimensional optimisation. We compare the COBYLA starting with 30 PCs and restarting with 100 PCs to the original COBYLA with 100 PCs. It can be observed that the number of objective function calls has increased, the objective value increased and the error norm decreased, all within the confi-



(a) Objective value

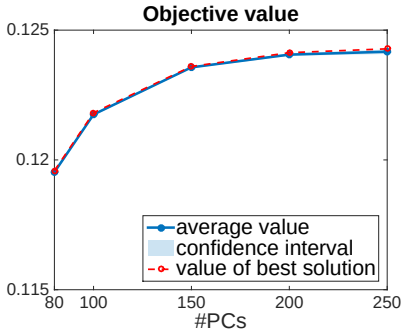


(b) Number of objective funcion calls made

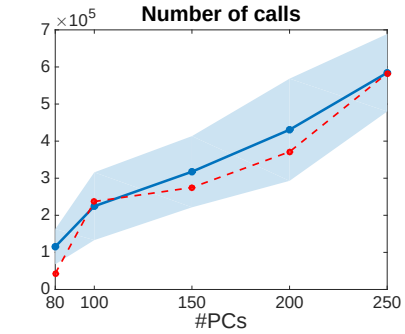


(c) Error norm with respect to SQP solution

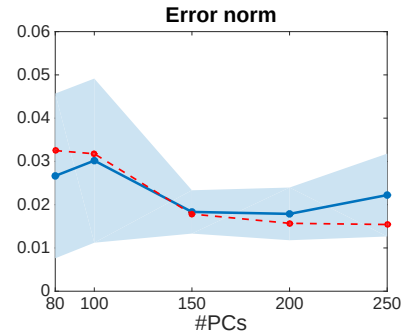
Figure 15: Best and average of 10 with confidence interval of 0.68 of COBYLA, using $\rho_{start} = 0.1$, $\rho_{end} = 10^{-4}$, $Re = 50$ and varying the number of PCs. The dots in (c) denote the individual runs.



(a) Objective value



(b) Number of objective function calls made



(c) Error norm with respect to SQP solution

Figure 16: Best and average of only the runs that count as successful in the ratio calculation, so that are in the correct optimum, with confidence interval of 0.68 of COBYLA, using $\rho_{start} = 0.1$, $\rho_{end} = 10^{-4}$, $Re = 25$ and varying the number of PCs.

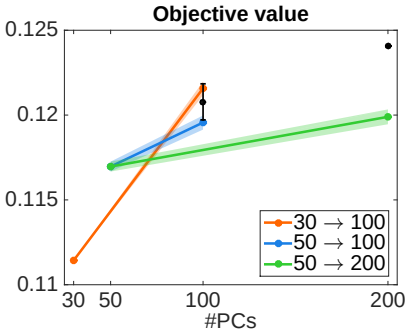
PC-PC	30-100	50-100	50-200
Ratio	1.0	0.0	0.0

Table 3: Ratio of correct solutions and number of total solutions of restarted COBYLA, at $Re = 25$, $\rho_{start} = 10^{-2}$, $\rho_{end} = 10^{-4}$ with varying dimensions.

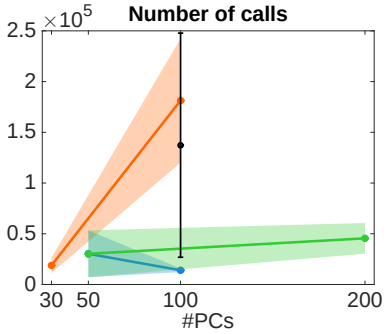
dence interval of the original 100 PC solution. However with a larger confidence interval, which means it is more robust. This is also what we conclude from the ratio, which is 1.0 instead of 0.5 for the original. The increase in objective function calls is not what we expected. However when we compare the number of objective function calls in restarted COBYLA to the number of objective function calls in Figure 16b, we see that the number of objective function calls is less for restarted COBYLA. The cause of this difference is that in general COBYLA has significantly less objective function calls when the global optimum is not obtained, which is the case in half of the runs with 100 PCs but not in any of the runs with restarted COBYLA using the 30 dimensional solution as initial point.

4.4 Particle Swarm Optimisation

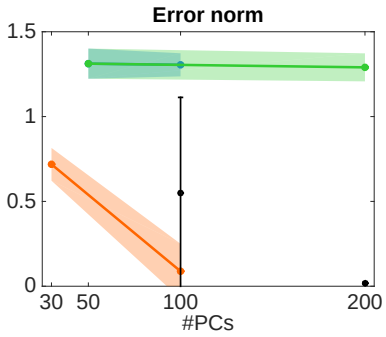
The PSO algorithm has been implemented using a swarm size of 100, 200, 400, 1000 and 2000. The number of iterations has been varied, up to 800. However, for these settings the algorithm was not able to converge when applied to the case with 100 PCs. Already the algorithm is at least 10 times as slow as COBYLA. So even if a good solution could be found using a larger swarm size or more iterations, COBYLA would still outperform PSO in terms of number of objective function calls. An explanation for the lack of results might be that wrong parameter settings have been used. While COBYLA has only two parameters that can be adapted (ρ_{start} and ρ_{end}), PSO has many more. While some studies have been done on the optimal parameter settings, for example in Li-Ping et al. [10], the optimal setting of most of the parameters depends on the problem that PSO is applied to. This means the parameters have



(a) Objective value



(b) Number of objective function calls made



(c) Error norm with respect to SQP solution

Figure 17: Best and average of 10 with confidence interval of 0.68 of COBYLA, using $\rho_{start} = 10^{-2}$, $\rho_{end} = 10^{-4}$, $Re = 25$ and using 100 PCs, with initial state the solution with 30 or 50 PCs, and 200 PCs with initial state the solution with 50 PCs. The black error bars are the solutions of COBYLA using 100 and 200 dimensions as seen in Figure 13.

to be tuned, which is itself an optimisation problem. This is both an advantage and a disadvantage. The advantage is that, when the parameters are tuned perfectly, the method is well adapted to solve your specific problem. However, when the parameters are not tuned well, the method might not be able to solve your problem at all, and finding the right parameter settings can be a difficult task. We can conclude from this that while there might exist parameter settings for PSO with which the algorithm might converge, or even outperform COBYLA, they are hard to find, and in that sense COBYLA is the favourable method.

5 Summary and discussion

The adjoint-free optimisation methods COBYLA and PSO have been applied to calculating CNOPs in the QG model. The performance of these methods has been tested, in order to compare the results to the performance of SQP, an adjoint-based optimisation method that is usually applied for calculating CNOPs. When testing the implementation of COBYLA, we have seen that for 100 PCs and $Re = 25$, COBYLA is reliable with $\rho_{end} = 10^{-5}$ or smaller. For $\rho_{end} = 10^{-4}$ COBYLA is robust for 150 PCs or more at $Re = 25$. There is no convergence yet with 200 PCs for $Re = 50$. This shows that for $Re = 25$ and $Re = 50$ the objective landscape and/or the dimension reduction is significantly different.

Restarted COBYLA is an improvement compared to normal COBYLA when a good initial solution is used for the restart, when the initial solution is not good, the solution is very bad. This means that restarted COBYLA is only suitable if the quality of the solution that is used as restart initial point, can be estimated. This premature conclusion could be validated by testing restarted COBYLA using the solutions of 100 and 150 principal components as initial points for a run in a higher dimension.

When comparing COBYLA to SQP in terms of performance, we see that for robust results COBYLA needs approximately 30,000 objective function calls, while SQP needs 200 to 600 calls to obtain the CNOPs to a much higher accuracy. So SQP outperforms COBYLA in terms of both accuracy and

speed. However COBYLA is able to find the CNOPs without the use of the adjoint model. A remark on the comparison of COBYLA and SQP is that the implementations of both methods use a slightly different implementation of the QG model, which means the comparison is slightly flawed. The objective values are not comparable, and the error norm as defined before is not expected to converge to zero but a small value above zero, which was observed to be close to 0.02. When convergence with more principal components is to be shown, this should be taken into account. Preferably the implementation of the objective function, including the model integration, is exactly the same for both methods. It should be noted that to draw more precise conclusions, the averages should be taken on more than 10 runs. For $Re = 50$ more than 200 PCs should be included to investigate whether the error norm can be reduced. Also it should be noted that the ratio that is included in the results, is a subjective measure and should, ideally, be replaced with a more objective measure of the resemblance of the global patterns of the CNOP.

Comparing our results to Mu et al. [11], we conclude that the amount of dimensions required is much larger for the QG model than for the ZC model: approximately 200 dimensions for a 2400 dimensional search space for the QG model, and only 30 dimensions for a 1080 dimensional search space for the ZC model. This is a possible explanation of why PSO works on the ZC model but not on the QG model. The large amount of dimensions required results in better objective values (in comparison to SQP) using COBYLA than in Mu et al. [11]. However, where their algorithm is only three times as slow as SQP, PCA based COBYLA is approximately 100 times as slow as SQP. The large difference in amount of PCs required to find the CNOPs in both models is remarkable.

At $Re = 25$ a divergence at 250 PCs can be observed. This shows that to demonstrate convergence when more dimensions are included, several higher dimensionalities have to be tested and included. However, the divergence might be suggesting that 250 dimensions could be close to the upper bound on the dimensionality COBYLA can solve. This means that if no convergence can be demonstrated, instead a more precise upper bound could be found. The question whether or not this slight divergence is a coincidence or indeed the upper bound of COBYLA is an important one, as it might limit the extent to which COBYLA can be applied to other models.

For the particle swarm the conclusion has been drawn that no convergence has been established, the explanation for this probably being that the parameter settings are wrong. A conclusion has been drawn that this wide range of parameters that have to be tuned is actually a flaw of the PSO algorithm, as it is a difficult task to find the right parameters. Of course COBYLA and PSO are just two examples of adjoint-free optimisation methods. Many other methods could be implemented on comparable models, to see whether they are suitable to resolve the CNOPs. Many of these models however suffer from the 'curse of dimensionality', which means it is hard to find a method that is known a priori to converge also on large models like general circulation models, as also the reduced space is larger than the one in this study. This shows however, that because many optimisation methods that could be applied to find CNOPs, suffer from the 'curse of dimensionality', which means that the dimension reduction is essential to solve these kind of problems. We have discussed two methods of dimension reduction and their advantages and disadvantages. Other methods of dimension reduction could also be considered, such as, in this case, Rossby basin modes. It should be taken into account however, that only (methods like) PCA can be applied to finding CNOPs of any quantity without adjusting the dimension reduction method. The challenge is to find a generic algorithm for dimension reduction that is able to resolve CNOPs of dynamical systems in low-dimensional spaces, even if the model dimensionality is very large.

As a next step, the COBYLA method could also be applied to models for which no adjoint model is available, as this study shows that COBYLA is able to solve the CNOP in a small system. We will discuss an example of a larger model, of which COBYLA might be able to solve the CNOPs. The Community Earth System Model [1] (CESM) is a fully coupled global atmosphere model and is freely available. For this model no adjoint model is available. We look at

the Coupled Model Intercomparison Project phase 5 configuration [2]. Using the 1.9x2.5_gx1v6 resolution, which yields a 2 degree grid in the land and atmospheric models and a 1 degree grid in the ocean and ice models, simulating 100 years would take approximately 24 hours (on 640 cores). Although very largely dependent on computing power, this number can be used as a rough estimate in translating the number of objective function calls to actual runtime. A physical property of which we could calculate the CNOP, is the meridional overturning circulation (MOC) dependence on sea surface salinity. This dependence can be estimated to be on the timescale of the order of decades. Therefore we look at T =10 years in this example. The search space is the size of the sea surface salinity component of the state vector. This is equal to the gridsize in horizontal direction in the sea model, which is $320 \times 384 = 122880$. This means the dimension of the search space is 122880. A rough estimate is that this could be reduced to 5000 to 6800 dimensions, using the same ratio as the reduction from 2400 to 100 to 150 as in our results. The applicability of this ratio is of course a very bold assumption. Under the assumption that COBYLA is still able to converge in approximately 6000 dimensions, which has not been shown yet, the amount of objective function calls, extrapolated from our results using 30 up to 250 principal components, a rough estimate would be that $2{\cdot}10^7$ functions calls are needed. One objective funcion call is 10 years of simulation, so the total simulation time is $2 \cdot 10^8$ years, which would on 640 cores lead to more than 5000 years of simulation. From this very rough estimate it can be concluded that even though calculating CNOPs of larger models might be possible using this method, further improvement of the algorithm would be necessary to calculate the CNOP of a larger model like the CESM as the amount of objective function calls is too large to solve the CNOP in any realistic amount of time.

Acknowledgements. I like to thank Henk en Tristan, for the supervision during my thesis, making this such an educational experience. Also I would like to thank Michael and Erik for helping me out so many times with my code. Special thanks to Roderick, Tjebbe, Brenda, René, Robby and my parents for the endless support during the last year.

References

[1] Community earth system model. URL `http://www2.cesm.ucar.edu/`.

[2] Coupled model intercomparison project phase 5. URL `http://www.wcrp-climate.org/wgcm-cmip/wgcm-cmip5`.

[3] Genevieve Coath and Saman K Halgamuge. A comparison of constraint-handling methods for the application of particle swarm optimization to constrained nonlinear optimization problems. In *Evolutionary Computation, 2003. CEC'03. The 2003 Congress on*, volume 4, pages 2419–2425. IEEE, 2003.

[4] Henk A Dijkstra. *Dynamical Oceanography*. Springer-Verlag Berlin Heidelberg, 2008.

[5] Henk A. Dijkstra. *Nonlinear climate dynamics*. Cambridge University Press, 2013.

[6] Andries P. Engelbrecht. *Heterogeneous Particle Swarm Optimization*, pages 191–202. Springer Berlin Heidelberg, Berlin, Heidelberg, 2010. ISBN 978-3-642-15461-4. doi: 10.1007/978-3-642-15461-4_17. URL `http://dx.doi.org/10.1007/978-3-642-15461-4_17`.

[7] A Hannachi. A primer for eof analysis of climate data. *Department of Meteorology, University of Reading*, pages 1–33, 2004.

[8] Eugenia Kalnay. *Atmospheric modeling, data assimilation and predictability*. Cambridge university press, 2003.

[9] Jeffrey C Lagarias, James A Reeds, Margaret H Wright, and Paul E Wright. Convergence properties of the nelder–mead simplex method in low dimensions. *SIAM Journal on optimization*, 9 (1):112–147, 1998.

[10] Zhang Li-Ping, Yu Huan-Jun, and Hu Shang-Xu. Optimal choice of parameters for particle swarm optimization. *Journal of Zhejiang University Science A*, 6(6):528–534, 2005.

[11] Bin Mu, Shicheng Wen, Shijin Yuan, and Hongyu Li. Ppso: Pca based particle swarm optimization for solving conditional nonlinear optimal perturbation. *Computers & Geosciences*, 83:65–71, 2015.

[12] Mu Mu. Nonlinear singular vectors and nonlinear singular values. *Science in China Series D: Earth Sciences*, 43(4):375–385, 2000. ISSN 1006-9313. doi: 10.1007/BF02959448. URL `http://dx.doi.org/10.1007/BF02959448`.

[13] Mu Mu, WS Duan, and Bin Wang. Conditional nonlinear optimal perturbation and its applications. *Nonlinear Processes in Geophysics*, 10(6): 493–501, 2003.

[14] Jorge Nocedal and Stephen Wright. *Numerical optimization*. Springer Science & Business Media, 2006.

[15] AR Osborne and A Pastorello. Simultaneous occurence of low-dimensional chaos and colored random noise in nonlinear physical systems. *Physics Letters A*, 181(2):159–171, 1993.

[16] Michael JD Powell. A direct search optimization method that models the objective and constraint functions by linear interpolation. In *Advances in optimization and numerical analysis*, pages 51–67. Springer, 1994.

[17] P.A. Simionescu. *Computer Aided Graphing and Simulation Tools for AutoCAD Users*. Boca Raton, FL: CRC Press, 2014.

[18] A. D. Terwisscha van Scheltinga and H. A. Dijkstra. Conditional nonlinear optimal perturbations of the double-gyre ocean circulation. *Nonlinear Processes in Geophysics*, 15(5):727–734, 2008. doi: 10.5194/npg-15-727-2008. URL `http://www.nonlin-processes-geophys.net/15/727/2008/`.

A Tables of Results

ρ_{end}	Best obj	Mean objective		Mean #Calls		Mean error norm	
10^{-2}	0.1142	0.1117	± 0.0024	3000	± 700	0.82	± 0.21
10^{-3}	0.1194	0.1187	± 0.0005	10000	± 3000	0.8	± 0.3
10^{-4}	0.1218	0.121	± 0.001	100000	± 100000	0.5	± 0.6
10^{-5}	0.1219	0.1218619	± 0.0000003	300000	± 100000	0.0164	± 0.0003

Table 4: Best objective value, average objective value, average number of objective function calls and average error norm with respect to SQP solution for $\rho_{start} = 0.1$, 100 PCs and varying ρ_{end}, for $Re = 25$.

#PC	Best obj	Mean objective		Mean #Calls		Mean error norm	
30	0.1115	0.11143	± 0.00002	19000	± 7600	0.7	± 0.1
50	0.1172	0.1170	± 0.0003	30000	± 23000	1.31	± 0.09
80	0.1196	0.1191	± 0.0004	61000	± 34000	0.6	± 0.4
100	0.1218	0.121	± 0.001	140000	± 110000	0.5	± 0.6
150	0.1236	0.12357	± 0.00003	320000	± 96000	0.018	± 0.005
200	0.1241	0.12406	± 0.00005	430000	± 140000	0.018	± 0.006
250	0.1243	0.12417	± 0.00009	580000	± 100000	0.02	± 0.01

Table 5: Best objective value, average objective value, average number of objective function calls and average error norm with respect to SQP solution for $\rho_{start} = 0.1$, $\rho_{end} = 10^{-4}$, $Re = 25$ and varying the number of PCs.

#PC	Best obj	Mean objective		Mean #Calls		Mean error norm	
50	0.1524	0.15236	± 0.00005	26000	± 7300	0.38	± 0.05
100	0.1668	0.1664	± 0.0004	55000	± 16000	0.42	± 0.05
200	0.1731	0.1729	± 0.0003	300000	± 120000	0.29	± 0.06

Table 6: Best objective value, average objective value, average number of objective function calls and average error norm with respect to SQP solution for $\rho_{start} = 0.1$, $\rho_{end} = 10^{-4}$, $Re = 50$ and varying the number of PCs.

PC-PC	Best obj	Mean objective		Mean #Calls		Mean error norm	
30-100	0.1218	0.1216	± 0.0004	200000	± 60000	0.09	± 0.16
50-100	0.1199	0.1196	± 0.0004	10000	± 2000	1.3	± 0.1
50-200	0.1203	0.1199	± 0.0004	50000	± 20000	1.3	± 0.1

Table 7: Best objective value, average objective value, average number of objective function calls and average error norm with respect to SQP solution for $\rho_{start} = 0.01$, $\rho_{end} = 10^{-4}$, $Re = 25$ and varying the number of PCs: using the 30 dimensional solution as initial point in 100 dimensions and using the 50 dimensional solution as initial point in 100 and 200 dimensions.

B Implementation

Here we will discuss which steps have to be taken to use COBYLA to calculate CNOPs in a dynamical system after dimension reduction. We assume that for a certain system a model integration function exists, and that we want to calculate a CNOP in a specific norm, with constraint boundary δ and integration time T, with respect to a certain background state $\boldsymbol{y}$.

B.1 PCA

For principal component analysis, we need the following steps.

1. Obtain a training set by running the model for a long period, at least 100 times T. The training set should be a matrix with columns consisting of states: each column is a time-sample of the integration.

2. Center the training set: subtract from each row the average of this row. This yields $\boldsymbol{X}$, the centered training set.

3. Calculate $\boldsymbol{X}\boldsymbol{X}^T$, and its eigenvalue decomposition. For example with NAG routine F01CKF (Mark 21)[2] and F02FCF (Mark 21).

4. Write the eigenvectors in a file, to be read by the optimisation program. Preferably the eigenvectors are written in direction of decreasing corresponding eigenvalue.

B.2 Working in reduced dimensionality

The search space used by COBYLA is of a reduced dimensionality. However, the model integration is done in the original dimension of the model. Therefore we need a translation from reduced space to the original space.

[2]Information on the NAG library can be found at `http://www.nag.co.uk/numeric/fl/manual/xhtml/mark21.xml`

1. Read the eigenvectors from the file. Read as many as the desired dimensionality in which the CNOPs are to be solved, say, the first k.

2. Write a function that projects a state $\boldsymbol{x}$ in the reduced dimensionality to a model state in the original dimension. The model state should be a sum $\boldsymbol{x}_m = \sum_{i=1}^{k} \boldsymbol{x}(i)\boldsymbol{a}_i$ where $\boldsymbol{x}$ is the reduced state, $\boldsymbol{a}_i$ is the ith eigenvector, and k is the reduced dimensionality.

B.3 COBYLA

To implement COBYLA and calculate the CNOPs, we need the following steps.

1. Download the COBYLA software (for Fortran) from `http://mat.uc.pt/~zhang/software.html#cobyla`. Example optimisation problems are included in the code.

2. Write an implementation of the norm in which the CNOP is to be calculated, which is a function of the state (in the original dimensionality) and returns the norm.

3. Write a function CALCFC of variable $\boldsymbol{x}$, to be used by COBYLA, in which the objective and constraint are calculated and returned.

   The constraint value $c(\boldsymbol{x})$ is calculated as $c(\boldsymbol{x}) = ||\boldsymbol{x}_m|| - \delta$ where the implementation of the norm is used, δ is the chosen constraint boundary, and $\boldsymbol{x}_m$ is the model state obtained from the reduced state.

   The objective value $J(\boldsymbol{x})$ is calculated by integrating the state $\boldsymbol{y} + \boldsymbol{x}_m$ for time T, integrating the state $\boldsymbol{y}$ for time T, subtracting the integrated states and calculating the norm of the result. Here $\boldsymbol{x}_m$ is again the model state obtained from the reduced state. Note that if $\boldsymbol{y}$ is a steady state, of course integrating $\boldsymbol{y}$ separately is not necessary.

4. Provide COBYLA with CALCFC, a value of ρ_{start} and ρ_{end}, an initial solution, the search space dimension and a maximum number of calls to CALCFC.