Numerical methods for studying transition probabilities in stochastic ocean-climate models

Baars, Sven

CHAPTER 2

BASIC CONCEPTS

In this chapter we discuss some of the basic concepts that will be used throughout this thesis. We start with Newton's method, which we use in both continuation methods and implicit time stepping methods, and then continue with iterative methods that we use for solving the linear systems that arise in Newton's method. After this, we describe the concept of bifurcation analysis and the pseudo-arclength continuation method which we use to perform this bifurcation analysis. Finally, we describe the equations that are used in our ocean model and discuss stochastic differential equations which we use to represent unresolved small-scale variability in the ocean.

2.1 Newton's method

Newton's method, or the Newton–Raphson method, is an iterative method for computing the roots of a function $F : \mathbb{R}^n \to \mathbb{R}^n$ (Atkinson, 1988). It can be derived directly from the Taylor series

$$F(\boldsymbol{x} + \Delta\boldsymbol{x}) = F(\boldsymbol{x}) + J(\boldsymbol{x})\Delta\boldsymbol{x} + \mathcal{O}(\Delta\boldsymbol{x}^2),$$

where $J(\boldsymbol{x})$ is the Jacobian of F. Assuming that $F(\boldsymbol{x}+\Delta\boldsymbol{x}) = 0$, and neglecting the second order term $\mathcal{O}(\Delta\boldsymbol{x}^2)$, we obtain a linear system

$$J(\boldsymbol{x})\Delta\boldsymbol{x} = -F(\boldsymbol{x}),$$

which can be solved for $\Delta\boldsymbol{x}$. $\boldsymbol{x} + \Delta\boldsymbol{x}$ will now be a better approximation of the actual root. Starting at some initial guess $\boldsymbol{x}_0$, we can apply this procedure iteratively, as is shown in Algorithm 1.

input: $\boldsymbol{x}_0$ Initial guess.
F, J Function for which we want to find a root and its Jacobian.
output: $\boldsymbol{x}_k$ Approximate solution.

1: **for** $i = 1, 2, \ldots$ until convergence **do**
2: **solve** $J(\boldsymbol{x}_{i-1})\Delta\boldsymbol{x}_i = -F(\boldsymbol{x}_{i-1})$
3: $\boldsymbol{x}_i = \boldsymbol{x}_{i-1} + \Delta\boldsymbol{x}_i$

Algorithm 1: Newton's method for computing the roots of $F(\boldsymbol{x})$.

2.2 Iterative methods

Linear systems as the one described in the previous section are often large and sparse. In this case, standard methods such as LU factorization with forward-backward substitution are often not efficient enough, and one instead resides to iterative methods (Van der Vorst, 2003). The general idea behind iterative methods is that we want to solve the system $A\boldsymbol{x} = \boldsymbol{b}$, and at each iteration i, we try to get a better approximate solution $\boldsymbol{x}_i$. We may also write this as $\boldsymbol{x} = \boldsymbol{x}_i + \boldsymbol{\epsilon}_i$, where $\boldsymbol{\epsilon}_i$ is the error at step i. Multiplication by A gives us

$$A\boldsymbol{\epsilon}_i = A(\boldsymbol{x} - \boldsymbol{x}_i) = \boldsymbol{b} - A\boldsymbol{x}_i.$$

Since we do not have the exact solution, we do not know the exact error either. Instead, we can try to solve the system

$$M\boldsymbol{z}_i = \boldsymbol{b} - A\boldsymbol{x}_i$$

for $\boldsymbol{z}_i$, with M an approximation of A that makes this system much easier to solve than the system $A\boldsymbol{x} = \boldsymbol{b}$. Now since M is an approximation of A, $\boldsymbol{z}_i$ is an approximation of the error. Thus solving the easier system leads to a better approximation of the solution: $\boldsymbol{x}_{i+1} = \boldsymbol{x}_i + \boldsymbol{z}_i$. The basic iteration introduced here, now leads to

$$\boldsymbol{x}_{i+1} = \boldsymbol{x}_i + M^{-1}(\boldsymbol{b} - A\boldsymbol{x}_i),$$

where M is called the *preconditioner*, which is used to improve the spectral properties of the system. Note that M^{-1} is never computed explicitly.

If we now take $M = I$, we obtain the well known Richardson iteration (Van der Vorst, 2003)

$$\boldsymbol{x}_{i+1} = \boldsymbol{b} + (I - A)\boldsymbol{x}_i = \boldsymbol{x}_i + \boldsymbol{r}_i,$$

with $\boldsymbol{r}_i = \boldsymbol{b} - A\boldsymbol{x}_i$ the residual at step i. Multiplying the above relation by $-A$ and adding $\boldsymbol{b}$ gives

$$\boldsymbol{r}_{i+1} = (I - A)\boldsymbol{r}_i = (I - A)^{i+1}\boldsymbol{r}_0.$$

It then follows that the approximate solution $\boldsymbol{x}_{i+1}$ may be written as

$$\boldsymbol{x}_{i+1} = \boldsymbol{r}_0 + \boldsymbol{r}_1 + \ldots + \boldsymbol{r}_i = \sum_{k=0}^{i} (I - A)^k \boldsymbol{r}_0 \tag{2.1}$$

for $\boldsymbol{x}_0 = 0$. We can do this without loss of generality, because in case $\boldsymbol{x}_0$ is nonzero, we could just shift the system by setting $A\boldsymbol{y} = \boldsymbol{b} - A\boldsymbol{x}_0 = \hat{\boldsymbol{b}}$ with $\boldsymbol{y}_0 = 0$. We now observe that

$$\boldsymbol{x}_{i+1} \in \mathrm{Span}\{\boldsymbol{r}_0, A\boldsymbol{r}_0, \ldots, A^i\boldsymbol{r}_0\} \equiv \mathcal{K}_{i+1}(A; \boldsymbol{r}_0).$$

The space of dimension m, spanned by a given vector $\boldsymbol{v}$, and increasing powers of A applied to $\boldsymbol{v}$ up to the $(m-1)$th power of A is called the m-dimensional *Krylov subspace* generated by A and $\boldsymbol{v}$, and is denoted as $\mathcal{K}_m(A; \boldsymbol{v})$ (Van der Vorst, 2003). In the general case that $M \neq I$, we obtain $\boldsymbol{x}_m \in \mathcal{K}_m(AM^{-1}; \boldsymbol{r}_0)$.

As we know from the power method, repeated multiplication by a matrix A converges to the eigenvector belonging to the largest eigenvalue of A. So (2.1) would converge to the eigenvector belonging to the largest eigenvalue of $I - A$. This means that this iteration without any modification is not a very good way to generate new vectors for the Krylov subspace.

Instead, one can orthogonalize every new vector with respect to all vectors that are already present in the Krylov subspace. This can be done by using for instance modified Gram-Schmidt (Golub and Van Loan, 1996). The power method with inclusion of the modified Gram-Schmidt procedure, which is given in Algorithm 2, is called the Arnoldi method. This method can generate an orthonormal basis of the Krylov subspace, but can also be used for the computation of multiple eigenvectors and eigenvalues of a matrix A.

With the orthonormal basis of the Krylov subspace V_m and the upper Hessenberg matrix $H_{m+1,m}$ that are obtained from the Arnoldi method, we get the relation

$$AV_m = V_{m+1}H_{m+1,m},$$

or alternatively

$$V_m^T A V_m = H_{m,m}.$$

The matrix $H_{m,m}$ is also called the *Galerkin projection* of A onto $\mathcal{K}_m(A; \boldsymbol{v}_1)$. For an iterative method based on this projection, the related *Galerkin condition* is given by

$$V_m^T(\boldsymbol{r}_0 - A\boldsymbol{x}_m) = V_m^T(\boldsymbol{r}_0 - AV_m\boldsymbol{y}_m) = 0$$

input:	A	Matrix.
	$\boldsymbol{v}_0$	Initial vector.
	m	Size of the Krylov subspace.
output:	$H_{m+1,m}$	Upper Hessenberg matrix.
	$V_m = [\boldsymbol{v}_1, \ldots, \boldsymbol{v}_m]$	Orthonormal basis of the m-dimensional Krylov subspace.

1: $\boldsymbol{v}_1 = \boldsymbol{v}_0 / \|\boldsymbol{v}_0\|_2$
2: **for** $j = 1, \ldots, m-1$ **do**
3: $\quad \boldsymbol{q} = A\boldsymbol{v}_j$
4: $\quad$ **for** $i = 1, \ldots, j$ **do**
5: $\quad\quad H_{i,j} = \boldsymbol{v}_i^T \boldsymbol{q}$
6: $\quad\quad \boldsymbol{q} = \boldsymbol{q} - H_{i,j}\boldsymbol{v}_i$
7: $\quad H_{j+1,j} = \|q\|_2$
8: $\quad \boldsymbol{v}_{j+1} = \boldsymbol{q} / H_{j+1,j}$

Algorithm 2: The Arnoldi method with modified Gram-Schmidt. V is an orthonormal basis of the m-dimensional Krylov subspace. The eigenvalues and eigenvectors of A can be obtained from the matrices $H_{m+1,m}$ and $V_m = [v_1, \ldots, v_m]$.

for some $\boldsymbol{x}_m = V_m \boldsymbol{y}_m + \boldsymbol{x}_0$ (Golub and Van Loan, 1996).

Most Krylov subspace methods can be distinguished in three different classes (Van der Vorst, 2003).

1. $\boldsymbol{x}_m$ is taken such that the residual is orthogonal to the Krylov subspace: $\boldsymbol{b} - A\boldsymbol{x}_m \perp \mathcal{K}_m(A; \boldsymbol{r}_0)$. This is called the *Ritz–Galerkin* approach.

2. $\boldsymbol{x}_m$ is taken such that the residual is orthogonal to some other suitable m-dimensional space. This is called the *Petrov–Galerkin* approach.

3. $\boldsymbol{x}_m$ is taken such that $\|\boldsymbol{b} - A\boldsymbol{x}_m\|_2$ is minimal over $\mathcal{K}_m(A; \boldsymbol{r}_0)$. This is called the *minimum residual norm* approach.

The most commonly used iterative methods are the Conjugate Gradient (CG) method for symmetric positive definite matrices, which is based on the Ritz–Galerkin approach, and the MINRES method for symmetric matrices and GMRES method for general matrices, which are based on the minimum residual norm approach.

The convergence behavior of Krylov subspace methods is determined by the spectral properties of A. Fast convergence is obtained when all eigenvalues of the matrix are close to 1, and when the matrix is close to a normal matrix, meaning that $A^T A = AA^T$. As mentioned above, a preconditioner can

be used to improve these properties. In Chapter 3 we introduce such a preconditioner for staggered grid discretizations of the Navier–Stokes equations.

2.3 Bifurcation analysis

As mentioned earlier, we are interested in the computation of tipping points in the MOC. In this section we discuss how we compute these tipping or bifurcation points.

Take a dynamical system of the form

$$\frac{\mathrm{d}\boldsymbol{x}}{\mathrm{d}t} = F(\boldsymbol{x}, \boldsymbol{p}), \tag{2.2}$$

where $\boldsymbol{x} \in \mathbb{R}^n$ is the state, $\boldsymbol{p} \in \mathbb{R}^m$ are parameter values and $F : \mathbb{R}^n \times \mathbb{R}^m \to \mathbb{R}^n$. We are interested in sets of solutions which are approached asymptotically when going forward or backward in time. These sets include periodic orbits, and stable and unstable steady states. A bifurcation is a point at which a small change of a so-called bifurcation parameter in $\boldsymbol{p}$ results in a sudden qualitative change of the system, e.g. when a stable steady state turns into an unstable steady state.

We can study bifurcations from a bifurcation diagram, in which a variable of interest $y : \mathbb{R}^n \times \mathbb{R} \to \mathbb{R}$ is plotted against the bifurcation parameter. An example of a bifurcation diagram is shown in Figure 2.1, where y is the variable of interest, and μ is the bifurcation parameter. The stable steady states are depicted by a solid line, whereas the unstable steady states are dashed. In the bifurcation diagram, we see two saddle-node bifurcations.

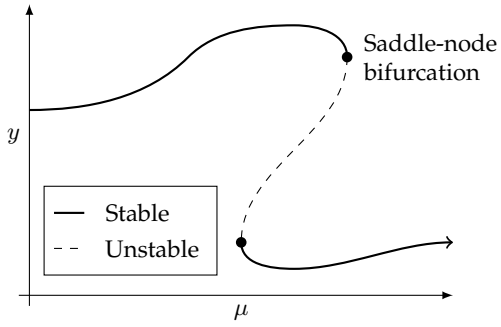

Figure 2.1: Schematic depiction of two saddle-node bifurcations, where at the first bifurcation a branch of stable steady states turns into a branch of unstable steady states, and at the second bifurcation an unstable branch turns into a stable branch. Here y is the variable of interest, and μ is the value of the bifurcation parameter.

The types of bifurcations that we will see in this thesis are saddle-node bifurcations, as shown in Figure 2.1, and pitchfork bifurcations, as shown in Figure 2.2.

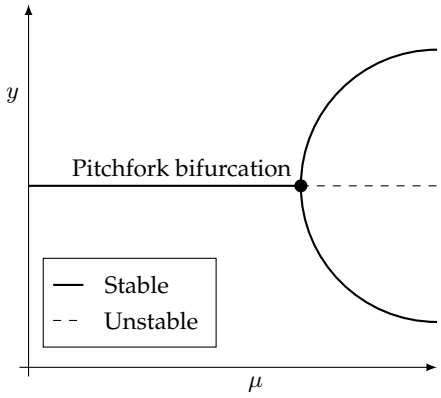

Figure 2.2: Schematic depiction of a supercritical pitchfork bifurcation, where a stable steady state branches into two stable steady states and one unstable steady state. Here y is the variable of interest, and μ is the value of the bifurcation parameter.

2.3.1 Pseudo-arclength continuation

For the study of bifurcation points in a dynamical system of the form (2.2), it is often desirable to compute steady states for a wide variety of parameter values $\boldsymbol{p}$. A common approach is to start from a known steady state at a nearby parameter value, and then perform a time integration to get to the stable steady state at the current parameter value.

A more efficient way of finding the steady state at a certain parameter value when starting from the steady state at a nearby parameter value is by computing the solution of

$$F(\boldsymbol{x}, \boldsymbol{p}) = 0 \tag{2.3}$$

directly by using Newton's method as described in Section 2.1. This is called natural continuation. Neither of these methods, however, is capable of the computation of unstable steady states after a saddle-node bifurcation is reached. The solution will just converge to one of the stable steady states instead.

Instead one can apply a method called pseudo-arclength continuation (Keller, 1977; Crisfield, 1991), which can speed up the computation of steady states considerably Dijkstra et al. (2014). In this method, the branches

$(\boldsymbol{x}(s), \boldsymbol{p}(s))$ in the bifurcation diagram are instead parameterized by an arclength parameter s. An additional normalizing condition

$$N(\boldsymbol{x}, \mu, s) \equiv \zeta \, \|\dot{\boldsymbol{x}}(s)\|_2^2 + (1-\zeta) \, |\dot{\mu}(s)|^2 - 1 = 0$$

is applied, where the dot indicates the derivative with respect to s, $\zeta \in (0,1)$ is a tunable parameter and μ is the bifurcation parameter in $\boldsymbol{p}$. This condition, however, is not very useful for both implementation or derivation of the method that we want to implement. Instead we reside to the approximation

$$N_2(\boldsymbol{x}, \mu, s) \equiv \zeta \, \|\boldsymbol{x}(s) - \boldsymbol{x}_0\|_2^2 + (1-\zeta) \, (\mu(s) - \mu_0)^2 - (s - s_0)^2 = 0. \tag{2.4}$$

Here $\Delta s = s - s_0$ is the step along the tangent vector at the current position $(\boldsymbol{x}_0, \boldsymbol{p}_0)$. In case $\dot{\boldsymbol{x}}_0$ and $\dot{\mu}_0$ are known, one can instead use the approximation

$$N_3(\boldsymbol{x}, \mu, s) \equiv \zeta \, \dot{\boldsymbol{x}}_0^T (\boldsymbol{x}(s) - \boldsymbol{x}_0) + (1-\zeta) \, \dot{\mu}_0 (\mu(s) - \mu_0) - (s - s_0) = 0.$$

To solve (2.3) together with (2.4), a predictor-corrector scheme is applied. The Euler predictor, which is a guess for the next steady state $\boldsymbol{x}_{i+1}$, is given by

$$\begin{aligned} \boldsymbol{x}_{i+1}^{(0)} &= \boldsymbol{x}_i + \Delta s \, \dot{\boldsymbol{x}}_i, \\ \mu_{i+1}^{(0)} &= \mu_i + \Delta s \, \dot{\mu}_i. \end{aligned}$$

In the Newton corrector, the updates are given by

$$\begin{aligned} \boldsymbol{x}^{(k+1)} &= \boldsymbol{x}^{(k)} + \Delta \boldsymbol{x}^{(k+1)}, \\ \mu^{(k+1)} &= \mu^{(k)} + \Delta \mu^{(k+1)}, \end{aligned}$$

where we dropped the $i+1$ subscript for readability. This process is shown in Figure 2.3.

$\Delta \boldsymbol{x}^{(k+1)}$ and $\Delta \mu^{(k+1)}$ can be obtained from the correction equation

$$\begin{pmatrix} J_{\boldsymbol{x}}(\boldsymbol{x}^{(k)}, \mu^{(k)}) & J_\mu(\boldsymbol{x}^{(k)}, \mu^{(k)}) \\ 2\zeta \, (\boldsymbol{x}^{(k)} - \boldsymbol{x}_i)^T & 2(1-\zeta) \, (\mu^{(k)} - \mu_i) \end{pmatrix} \begin{pmatrix} \Delta \boldsymbol{x}^{(k+1)} \\ \Delta \mu^{(k+1)} \end{pmatrix} = \begin{pmatrix} -F(\boldsymbol{x}^{(k)}, \mu^{(k)}) \\ -N_2(\boldsymbol{x}^{(k)}, \mu^{(k)}, s) \end{pmatrix},$$

where $J_{\boldsymbol{x}}$ is the Jacobian of F with respect to $\boldsymbol{x}$ and J_μ is the Jacobian of F with respect to μ. This system can be solved as such, which is very robust, but has the downside that a solver for this bordered system has to be implemented.

The solution can also be obtained by first solving

$$\begin{aligned} J_{\boldsymbol{x}}(\boldsymbol{x}^{(k)}, \mu^{(k)}) \boldsymbol{z}_1 &= -F(\boldsymbol{x}^{(k)}, \mu^{(k)}), \\ J_{\boldsymbol{x}}(\boldsymbol{x}^{(k)}, \mu^{(k)}) \boldsymbol{z}_2 &= J_\mu(\boldsymbol{x}^{(k)}, \mu^{(k)}), \end{aligned}$$

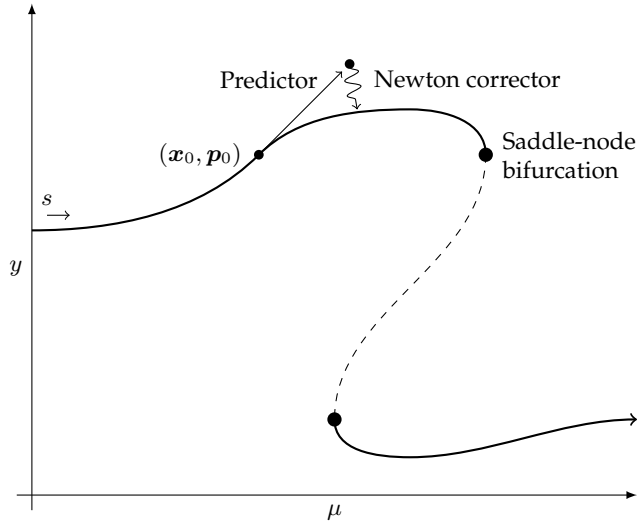

Figure 2.3: Schematic depiction of pseudo-arclength continuation.

after which $\Delta\boldsymbol{x}^{(k+1)}$ and $\Delta\mu^{(k+1)}$ can be found from

$$\begin{aligned}\Delta\mu^{(k+1)} &= \frac{-N_2(\boldsymbol{x}^{(k)}, \mu^{(k)}, s) - 2\zeta\ (\boldsymbol{x}^{(k)} - \boldsymbol{x}_i)^T \boldsymbol{z}_1}{2(1-\zeta)\ (\mu^{(k)} - \mu_i) - 2\zeta\ (\boldsymbol{x}^{(k)} - \boldsymbol{x}_i)^T \boldsymbol{z}_2},\\ \Delta\boldsymbol{x}^{(k+1)} &= \boldsymbol{z}_1 - \Delta\mu^{(k+1)} \boldsymbol{z}_2.\end{aligned}$$

This method is less robust, but has the advantage that only systems with $J_{\boldsymbol{x}}$ have to be solved.

Linear approximations to the derivatives $\dot{\boldsymbol{x}}$ and $\dot{\mu}$, which are required for the Euler predictor, can be obtained from two previous iterations of the method as

$$\begin{aligned}\dot{\boldsymbol{x}}_i &\approx \frac{1}{\Delta s}(\boldsymbol{x}_i - \boldsymbol{x}_{i-1}),\\ \dot{\mu}_i &\approx \frac{1}{\Delta s}(\mu_i - \mu_{i-1}).\end{aligned}$$

This, however, can not be used in the first step. What we do instead is first obtain $\boldsymbol{x}_0$ at parameter μ_0 by using Newton's method, and then compute the tangent by solving

$$J_{\boldsymbol{x}}(\boldsymbol{x}_0, \mu_0)\boldsymbol{z} = J_\mu(\boldsymbol{x}_0, \mu_0)$$

for $\boldsymbol{z}$, after which

$$\dot{\boldsymbol{x}}_0 \approx -\frac{1}{\sqrt{\boldsymbol{z}^T\boldsymbol{z} + \Delta\mu^2}}\boldsymbol{z},$$
$$\dot{\mu}_0 \approx \frac{\Delta\mu}{\sqrt{\boldsymbol{z}^T\boldsymbol{z} + \Delta\mu^2}},$$

where we take $\Delta\mu = 1$.

2.4 Stochastic differential equations

Stochastic differential equations (SDEs) are used in a wide variety of applications areas including biology, chemistry, physics, economics, and of course climate dynamics. In this thesis, we will use them to represent unresolved variability in the freshwater forcing. We only briefly introduce the concepts that are required for understanding the usage of SDEs in this thesis. For a more detailed explanation, we refer to Higham (2001), Gardiner (1985) or Dijkstra (2013).

2.4.1 Brownian motion

A *standard Brownian motion* or *Wiener process* is a random variable W_t on $t \in [0, T]$ which has the following properties:

1. $W_0 = 0$ with probability 1.
2. W has independent increments $W_t - W_s$ for every $t > s \geq 0$.
3. For $0 \leq s < t \leq T$, the increment $W_t - W_s$ is normally distributed with mean zero and variance $t - s$, i.e. $W_t - W_s \sim N(0, t - s)$
4. W_t is continuous in t with probability 1.

In a computational context, we always look at discretized Brownian motion. This means that the value of W_t is only specified at discrete values of t. Say we take a fixed number of time steps N with length $\Delta t = T/N$, then the discretized Brownian motion would be defined as

$$W_i = W_{i-1} + \Delta W_i,$$

where W_i is the value of W_t at time $t_i = i\Delta t$, $i = 0, 1, \ldots, N$ and ΔW_i is the increment at time t_i which is an independent random variable taken from $N(0, \Delta t)$. An example of a discretized Brownian motion can be found in Figure 2.4.

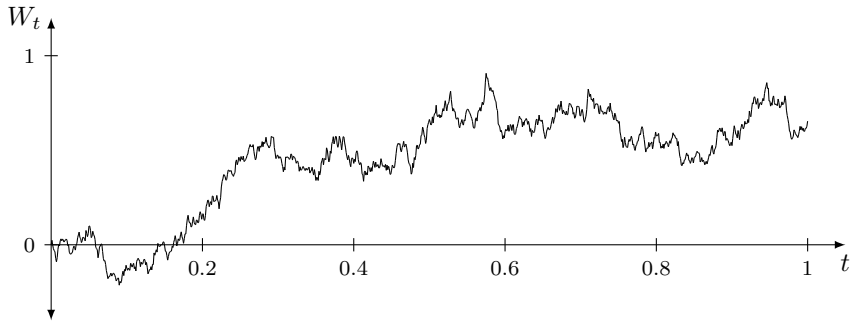

Figure 2.4: Example of a single realization of a one-dimensional discretized Brownian motion.

2.4.2 Stochastic differential equations

Given a real function $f : [0, T] \to \mathbb{R}$, we approximate the integral $\int_0^T f(t)\mathrm{d}t$ by the left Riemann sum

$$\sum_{i=0}^{N-1} f(t_i)(t_{i+1} - t_i), \tag{2.5}$$

where $t_i = i\Delta t$ with $\Delta t = \frac{T}{N}$. For Riemann-integrable functions, (2.5) will converge to the actual value of the integral for $\Delta t \to 0$.

Similarly, we may consider an approximation of the form

$$\sum_{i=0}^{N-1} g(t_i)(W_{i+1} - W_i),$$

for a stochastic integral $\int_0^T g(t)\mathrm{d}W_t$. If we again take $\Delta t \to 0$, this defines the value of the so-called *Itô integral*.

For stochastic integration, one may also use different approximations to obtain a different integral value. After the Itô integral, the most common integral is the Stratonovich integral, which instead of using a left Riemann-type sum uses the midpoint rule. In this thesis, however, we will only use Itô integration and the related Itô calculus.

Using the Itô integral, we may define a stochastic differential equation in integral form

$$X_t = X_0 + \int_0^t f(X_s)\mathrm{d}s + \int_0^t g(X_s)\mathrm{d}W_s, \quad 0 \le t \le T,$$

where the initial condition X_0 is a random variable, and f and g are scalar functions. In differential equation form, this integral is given by

$$\mathrm{d}X_t = f(X_t)\mathrm{d}t + g(X_t)\mathrm{d}W_t, \quad 0 \le t \le T.$$

The ocean can, unfortunately, not be described by a single stochastic differential equation. We will instead look at systems of stochastic differential algebraic equations (SDAEs) of the form

$$M\,\mathrm{d}\boldsymbol{X}_t = F(\boldsymbol{X}_t)\,\mathrm{d}t + g(\boldsymbol{X}_t)\,\mathrm{d}\boldsymbol{W}_t, \tag{2.6}$$

where $M \in \mathbb{R}^{n\times n}$ is usually referred to as the mass matrix, $\boldsymbol{X}_t \in \mathbb{R}^n$, $F : \mathbb{R}^n \to \mathbb{R}^n$, $g : \mathbb{R}^n \to \mathbb{R}^{n_w}$, $\boldsymbol{W}_t \in \mathbb{R}^{n_w}$, and n_w is the number of stochastic variables.

2.4.3 The Euler–Maruyama method

The most basic method for simulating stochastic differential equations in Itô calculus is the Euler–Maruyama method, which is a generalization of the Euler method for ordinary differential equations Higham (2000). Consider the n-dimensional SDE

$$\mathrm{d}\boldsymbol{X}_t = F(\boldsymbol{X}_t)\,\mathrm{d}t + g(\boldsymbol{X}_t)\,\mathrm{d}\boldsymbol{W}_t.$$

The Euler–Maruyama method is then given by

$$\boldsymbol{X}_{i+1} = \boldsymbol{X}_i + \Delta t F(\boldsymbol{X}_i) + g(\boldsymbol{X}_i)\,\Delta\boldsymbol{W}_i. \tag{2.7}$$

where $\Delta\boldsymbol{W}_i$ are independent and identically distributed random variables with mean zero and variance Δt.

2.4.4 The stochastic theta method

In the same sense that the Euler method can be generalized to the theta method for ordinary differential equations, we can also generalize the Euler–Maruyama method (Kloeden and Platen, 1992). The generalization of this method is given by

$$\boldsymbol{X}_{i+1} = \boldsymbol{X}_i + (1-\theta)\Delta t\; F(\boldsymbol{X}_i) + \theta\Delta t\; F(\boldsymbol{X}_{i+1}) + g(\boldsymbol{X}_i)\,\Delta\boldsymbol{W}_i.$$

We will refer to this method as the stochastic theta method, as is also done in Higham (2000). Note that if we take $\theta = 0$, we get the Euler–Maruyama scheme as described above, with $\theta = 1$, we get the stochastic version of the implicit Euler method, and $\theta = \frac{1}{2}$ gives a stochastic trapezoidal rule. In any case, the noise is always added explicitly.

This method can be generalized even further for SDAEs, in which case it is given by

$$M(\boldsymbol{X}_i - \boldsymbol{X}_{i+1}) + (1-\theta)\Delta t\; F(\boldsymbol{X}_i) + \theta\Delta t\; F(\boldsymbol{X}_{i+1}) + g(\boldsymbol{X}_i)\,\Delta\boldsymbol{W}_i = 0. \tag{2.8}$$

Here $\boldsymbol{X}_{i+1}$ can be obtained by applying the Newton method. The Jacobian of (2.8) is given by

$$\theta\Delta t J(\boldsymbol{X}_{i+1}) - M,$$

where J is the Jacobian of F.

2.5 Governing equations

2.5.1 The Navier–Stokes equations

The Navier–Stokes equations, which govern the motion of a fluid such as the ocean, are described by the conservation of mass

$$\frac{\partial \rho}{\partial t} + \nabla \cdot (\rho \boldsymbol{u}) = 0,$$

and the conservation of momentum

$$\rho \frac{\partial \boldsymbol{u}}{\partial t} + \rho \boldsymbol{u} \cdot \nabla \boldsymbol{u} = -\nabla p + \nabla \cdot \boldsymbol{\tau},$$

where $\boldsymbol{u}$ is the velocity, p the pressure, ρ the density and $\boldsymbol{\sigma} = -p\boldsymbol{I} + \boldsymbol{\tau}$ is the internal stress tensor (Batchelor, 2000; Landau and Lifshitz, 1959).

In case the fluid is assumed to be incompressible, i.e. the density is constant, the conservation of mass is reduced to

$$\nabla \cdot \boldsymbol{u} = 0.$$

Furthermore, the stress tensor can be simplified and is now given by

$$\boldsymbol{\tau} = \mu(\nabla \boldsymbol{u} + \nabla \boldsymbol{u}^T),$$

where μ is the dynamic viscosity. In this case the conservation of momentum can be written as

$$\frac{\partial \boldsymbol{u}}{\partial t} + \boldsymbol{u} \cdot \nabla \boldsymbol{u} = -\frac{1}{\rho} \nabla p + \nu \nabla^2 \boldsymbol{u},$$

where $\nu = \mu / \rho$ is the kinematic viscosity.

2.5.2 The ocean model

Due to the complex nature of the Navier–Stokes equations, many approximations are made to be able to focus on the most important motions that are present in the ocean (Griffies, 2004; Cushman-Roisin and Beckers, 2011). The most important approximations are as follows. First we have the hydrostatic approximation, where the contributions of vertical acceleration and friction to the vertical pressure gradients are neglected. Then we have the shallow ocean approximation, where the depth of the ocean is assumed to be negligible with respect to the radius of the earth. The third approximation is the Boussinesq approximation, which means that any density difference that is not multiplied by the gravitational constant g is ignored. And finally we have the rigid lid approximation which assumes that the sea surface elevation is static.

For our simulations, this results in the spatially quasi two-dimensional primitive equation model of the Atlantic MOC as described in Weijer and Dijkstra (2001); Den Toom et al. (2011). In the model, there are two active tracers: temperature T and salinity S, which are related to the density ρ by a linear equation of state

$$\rho = \rho_0 \left(1 - \alpha_T \left(T - T_0\right) + \alpha_S \left(S - S_0\right)\right),$$

where α_T and α_S are the thermal expansion and haline contraction coefficients, respectively, and ρ_0, T_0, and S_0 are reference quantities. The numerical values of the fixed model parameters are summarized in Table 2.1.

D	=	4.0	$\cdot$ 10^3	m	H_m	=	2.5	$\cdot$ 10^2	m
r_0	=	6.371	$\cdot$ 10^6	m	T_0	=	15.0		°C
g	=	9.8		m s^{-2}	S_0	=	35.0		psu
A_H	=	2.2	$\cdot$ 10^{12}	m^{2}s^{-1}	α_T	=	1.0	$\cdot$ 10^{-4}	K^{-1}
A_V	=	1.0	$\cdot$ 10^{-3}	m^{2}s^{-1}	α_S	=	7.6	$\cdot$ 10^{-4}	psu^{-1}
K_H	=	1.0	$\cdot$ 10^{3}	m^{2}s^{-1}	ρ_0	=	1.0	$\cdot$ 10^{3}	kg m^{-3}
K_V	=	1.0	$\cdot$ 10^{-4}	m^{2}s^{-1}	τ	=	75.0		days

Table 2.1: Fixed model parameters of the two-dimensional ocean model.

In order to eliminate longitudinal dependence from the problem, we consider a purely buoyancy-driven flow on a non-rotating Earth. We furthermore assume that inertia can be neglected in the meridional momentum equation. The mixing of momentum and tracers due to eddies is parameterized by simple anisotropic diffusion. In this case, the zonal velocity as well as the longitudinal derivatives are zero and the equations for the meridional velocity v, vertical velocity w, pressure p, and the tracers T and S are given by

$$\begin{aligned}
-\frac{1}{\rho_0 r_0}\frac{\partial p}{\partial \theta} + A_V \frac{\partial^2 v}{\partial z^2} + \frac{A_H}{r_0^2}\left(\frac{1}{\cos\theta}\frac{\partial}{\partial\theta}\left(\cos\theta\frac{\partial v}{\partial\theta}\right) + \left(1-\tan^2\theta\right)v\right) &= 0,\\
-\frac{1}{\rho_0}\frac{\partial p}{\partial z} + g\left(\alpha_T T - \alpha_S S\right) &= 0,\\
\frac{1}{r_0\cos\theta}\frac{\partial v\cos\theta}{\partial\theta} + \frac{\partial w}{\partial z} &= 0,\\
\frac{\partial T}{\partial t} + \frac{v}{r_0}\frac{\partial T}{\partial\theta} + w\frac{\partial T}{\partial z} = \frac{K_H}{r_0^2\cos\theta}\frac{\partial}{\partial\theta}\left(\cos\theta\frac{\partial T}{\partial\theta}\right) + K_V\frac{\partial^2 T}{\partial z^2} &+ \mathrm{CA}(T),\\
\frac{\partial S}{\partial t} + \frac{v}{r_0}\frac{\partial S}{\partial\theta} + w\frac{\partial S}{\partial z} = \frac{K_H}{r_0^2\cos\theta}\frac{\partial}{\partial\theta}\left(\cos\theta\frac{\partial S}{\partial\theta}\right) + K_V\frac{\partial^2 S}{\partial z^2} &+ \mathrm{CA}(S).
\end{aligned}$$

Here t is time, θ latitude, z the vertical coordinate, r_0 the radius of Earth, g the acceleration due to gravity, A_H (A_V) the horizontal (vertical) eddy viscosity,

and K_H (K_V) the horizontal (vertical) eddy diffusivity. The terms with CA represent convective adjustment contributions.

The equations are solved on an equatorially symmetric, flat-bottomed domain. The basin is bounded by latitudes $\theta = -\theta_N$ and $\theta = \theta_N = 60°$ and has depth D. Free-slip conditions apply at the lateral walls and at the bottom. Rigid lid conditions are assumed at the surface and atmospheric pressure is neglected. The wind stress is zero everywhere, and "mixed" boundary conditions apply for temperature and salinity,

$$\begin{aligned} K_V \frac{\partial T}{\partial z} &= \frac{H_m}{\tau} \left(\bar{T}(\theta) - T \right), \\ K_V \frac{\partial S}{\partial z} &= S_0 F_s(\theta). \end{aligned} \tag{2.9}$$

This formulation implies that temperatures in the upper model layer (of depth H_m) are relaxed to a prescribed temperature profile $\bar{T}$ at a rate τ^{-1}, while salinity is forced by a net freshwater flux F_s, which is converted to an equivalent virtual salinity flux by multiplication with S_0. The specification of the CA terms is given in Den Toom et al. (2011).

2.5.3 Stochastic freshwater forcing

For the most simple deterministic case, we choose an equatorially symmetric surface forcing as

$$\bar{T}(\theta) = 10.0 \cos\left(\pi\theta/\theta_N\right), \tag{2.10a}$$

$$F_s(\theta) = \bar{F}_s(\theta) = \mu F_0 \frac{\cos(\pi\theta/\theta_N)}{\cos(\theta)}, \tag{2.10b}$$

where μ is the strength of the mean freshwater forcing (which we take as bifurcation parameter) and $F_0 = 0.1$ m yr^{-1} is a reference freshwater flux.

To represent the unresolved variability in the freshwater forcing, we use a stochastic forcing. This forcing is chosen as

$$F_s(\theta, t) = (1 + \sigma\, \zeta(\theta, t))\bar{F}_s(\theta), \tag{2.11}$$

where $\zeta(\theta, t)$ represents zero-mean white noise with a unit standard deviation, i.e., with $\mathbb{E}[\zeta(\theta, t)] = 0$, $\mathbb{E}[\zeta(\theta, t)\zeta(\theta, s)] = \delta(\theta, t - s)$ and $\mathbb{E}[\zeta(\theta, t)\zeta(\eta, t)] = \delta(\theta - \eta, t)$. The constant σ is the standard deviation of the noise which we set to $\sigma = 0.1$. The noise is additive and is only active in the freshwater component, only present at the surface, meridionally uncorrelated, and has magnitude σ of 10% of the background freshwater forcing amplitude at each latitude θ. In the context of (2.6), g is given by the discretized version of $\sigma\bar{F}_s(\theta)$.