Quantifying Uncertainty in Biogeochemical Models using Stochastic Collocation

J. Jakeman

Department of Mathematics
The Australian National University
Motivation
Biogeochemical Modelling

Figure: MPB = Microphytobenthos, DN = Dissolved Nutrients
Biogeochemical Modelling

Stochastic Collocation
Jakeman, John

Introduction
Estuarine Modelling
Problem Definition
Generalised Polynomial Chaos
Stochastic Collocation
Biogeochemical Modelling

Stochastic Collocation

Jakeman, John

Introduction

Estuarine Modelling

Problem Definition

Generalised Polynomial Chaos

Stochastic Collocation
Patterns in Observed Data
A Graphical Model
A Simple Model

\[
\frac{dg}{dt} = g \left[ r_g \left( \frac{1}{1 + q_g(k_g g + k_b b)} \right) \frac{P_b - g}{h + P - b - g} - d_g - f \right]
\]

\[
\frac{db}{dt} = b \left[ r_b \left( \frac{1}{1 + q_b(k_g g + k_b b)} \right) \frac{P_b - g}{h + P - b - g} - d_b - f \right]
\]
Figure: Bifurcation diagram of the model showing for which combinations of flush rate ($f$) and total $P$ concentration, blue-green algae ($b$), or other algae ($g$), will dominate, and for which combinations those states are alternative equilibria ($g/b$).
A Simple Model

Figure: Hysteresis shown as the response of the turbidity \( E \) with respect to the control parameters total P concentration \( P \) and flush rate \( f \)
We will focus our attention on the general differential equation

\[ \mathcal{L}(x, t, p; u) = f(x, t, p) \quad \forall x \in D, t \in (t_0, T] \]

\[ \mathcal{B}(x, t, p; u) = g(x, t, p) \quad \forall x \in \partial D, t \in (t_0, T] \]

Define probability space \((\Omega, \mathcal{F}, \mathcal{P})\). Want to find stochastic solution

\[ u : \Omega \times D \rightarrow \mathbb{R} \]
Probabilistic Framework

- Finite dimensional noise assumption:
  Model noise as $N$-variate random vector

  $$\xi(\omega) = (\xi_1(\omega), \ldots, \xi_N(\omega))$$

- Assume model inputs are functions of these random variables
  $$p(\omega) = p(\xi)$$

- Independence: $\xi_i(\omega)$ are independent random variables with probability density functions $\rho_i : \Gamma_i \rightarrow \mathbb{R}^+$ and joint density

  $$\rho(\xi) = \prod_{i=1}^{N} \rho_i(\xi_i) \quad \xi \in \Gamma = \prod_{i=1}^{n} \Gamma_i \subset \mathbb{R}^N$$
Parametric uncertainty: Model parameters as an $N$-variate random vector

$$\mathbf{p} = (p_1(\omega), \ldots, p_j(\omega)), \quad j = N$$

Stochastic Processes: Represent stochastic fields $p_j(x, t, \omega)$ by a finite-term Karhunen-Loève (KL) Stochastic Process

$$p_j(x, t, \omega) = E(p_j)(x, t) + \sum_{i=1}^{N} \sqrt{\lambda_i} \phi_i(x, t) \cdot \xi_i(\omega)$$

- Determin. Functs.
- Mean
- Rand. Var.
Generalised Polynomial Chaos (gPC): Definition

- The general second-order random process \( Y(x, t, \omega) \) can be represented by

\[
Y(x, t, \xi) = \sum_{i=0}^{\infty} y_i(x, t) \Phi_i(\xi)
\]

- \( \Phi_i(\xi) \) forms a complete orthogonal basis with respect to the inner product

\[
\langle f(\xi)g(\xi) \rangle = \int_{\Gamma} f(\xi)g(\xi)\rho(\xi)d\xi
\]

- Coefficients given by

\[
y_i = \frac{\langle Y(x, t, \xi)\Phi_i(\xi) \rangle}{\langle \Phi_i^2(\xi) \rangle}
\]
Limitations of gPC

- Computational complexity increases rapidly with the number of random inputs and the order of the expansion.

\[ P + 1 = \frac{(N + M)!}{N!M!} \]
Limitations of gPC

- Computational complexity increases rapidly with the number of random inputs and the order of the expansion.
- A high expansion order is required when the dependence of the solution on the random input data varies rapidly or if a singularity exists in the random space.
Figure: 10 term gPC approximation of the solution to the 1D Kraichnan-Orszag Problem
Limitations of gPC

- Computational complexity increases rapidly with the number of random inputs and the order of the expansion.
- A high expansion order is required when the dependence of the solution on the random input data varies rapidly or if a singularity exists in the random space.
- Errors in the gPC approximation of transient solutions may become unacceptably large even after only a short time.
Example: \[ \frac{dy(t, \xi)}{dt} = k(\xi)y(t, \xi), \quad y(0) = \hat{y} \]

**Figure:** Long-term evolution of mean and variance using a 5-th order expansion

**Figure:** Error convergence of the mean and variance at $t = 1$ and $t = 10$
Limitations of gPC

- Computational complexity increases rapidly with the number of random inputs and the order of the expansion.
- A high expansion order is required when the dependence of the solution on the random input data varies rapidly or if a singularity exists in the random space.
- Errors in the gPC approximation of transient solutions may become unacceptably large even after only a short time.
- gPC necessitates the solution of a system of coupled equations that require efficient and robust solvers and modification of deterministic code.
SC seeks to approximate the solution using Lagrange interpolation

\[ u(x, t, \xi) = \sum_{k=0}^{m} u_k(x, t, \xi_k) L_k(\xi) \]

where \( u_k(x, t, \xi_k) \) is the solution at the collocation points \( \{\xi_k\}_{k=0}^{m} \) and \( L_k \) is the Lagrange polynomial of order \( m + 1 \) that satisfies the usual relation \( L_k(\xi_i) = \delta_{ik} \).

\[ L_k(x) = \prod_{j=0, j \neq k}^{m} \frac{x - x_j}{x_k - x_j} \]
We wish to minimise the error in the SC approximation using the Galerkin method.

\[
\int_{\Gamma} \mathcal{L}(x, t, \xi; u) \nu(\xi) \rho(\xi) d\xi = \int_{\Gamma} f(x, t, \xi) \nu(\xi) \rho(\xi) d\xi \quad \forall \nu(\xi) \in V_{\Gamma}
\]

where \( V_{\Gamma} = \{ L_k \}_{k=0}^{m} \subset L_2(\Omega, \mathcal{F}, \mathcal{P}) \) By orthogonality property of the Lagrange polynomials

\[
\mathcal{L}(x, t, \xi_k; u) = f(x, t, \xi_k)
\]
Tensor Product Spaces

Without loss of generality consider $u \in [0,1]^N$. For $N = 1$ we can approximate $u$ using the one-dimensional interpolation formula

$$U^i(u) = \sum_{k=0}^{m^i} u(\xi_k^i) \cdot a^i_k$$

In the multivariate case $N > 1$ we can approximate $u$ by

$$(U^1 \otimes \cdots \otimes U^N) = \sum_{k_1=0}^{m^1} \cdots \sum_{k_N=0}^{m^N} u(\xi_{k_1}^1, \ldots, \xi_{k_N}^N) \cdot (a^1_{k_1} \otimes \cdots \otimes a^N_{k_N})$$

based upon the collocation sets

$$X^i = (\xi_1^i, \ldots, \xi_{m^i}^i)$$
The tensor product approach suffers curse of dimensionality so use more efficient sparse grid construction (Smolyak algorithm)

\[ A_{q,N}(u) = \sum_{q-N+1 \leq |i| \leq q} (-1)^{q-|i|} \cdot \binom{N-1}{q-|i|} \cdot (U_{i_1} \otimes \cdots \otimes U_{i_N}) \]

where \( i = (i_1, \ldots, i_N) \) and \(|i| = i_1 + \cdots + i_N\) To compute \( A_{q,N}(u) \) only need to evaluate function \( u \) on the “sparse grid”

\[ H_{q,N} = \bigcup_{q-N+1 \leq |i| \leq q} (X^{i_1} \times \cdots \times X^{i_N}) \]
Limitations of Traditional Approach

**Figure:** Traditional Sparse Grid Approximation of the hat function using Lagrange of varying order
Sparse Grids

It is sometimes easier to consider the incremental interpolant

\[ \Delta^i = u^i - u^{i-1}, \quad u^0 = 0 \]

The sparse grid interpolant is then just

\[
A_{q,N}(u) = \sum_{|i| \leq q} (\Delta^{i_1} \otimes \cdots \otimes \Delta^{i-N}) = A_{q-1,N}(u) + \Delta A_{q,N}(u)
\]

where

\[
A_{N-1,N}(u) = 0, \quad \text{and} \quad \Delta A_{q,N}(u) = \sum_{|i| = q} (\Delta^{i_1} \otimes \cdots \otimes \Delta^{i-N})
\]
Sparse Grids

In 1D choose nested collocation nodes $X^i$ such that

$$X^i \subset X^{i+1}$$

(1)

So on the sparse grid we can go from an order $q - 1$ interpolation to an order $q$ interpolation by simply evaluating the function $u$ at the points

$$\Delta H_{q,N} = \bigcup_{|i|=q} (X^i_\Delta \times \cdots \times X^N_\Delta)$$

where $X^i_\Delta = X^i \setminus X^{i-1}$
Choice of Collocation Points and Hierarchical Basis

In 1D choose the following collocation points

\[ m^i = \begin{cases} 
1 & \text{if } i = 1 \\
2^{i-1} + 1 & \text{if } i > 0 
\end{cases} \]

\[ \xi^i_k = \begin{cases} 
\frac{k-1}{m^i-1} & \text{for } k = 1, \ldots, m^i \text{ if } m^i > 1 \\
0.5 & \text{if } k = 1, \text{ if } m^i = 1 
\end{cases} \]

and piecewise linear basis functions

\[ a^i_k(\xi) = \begin{cases} 
1 - (m^i - 1)|\xi - \xi^i_k| & \text{if } |\xi - \xi^i_k| < 1/(m^i - 1) \\
0 & \text{otherwise} 
\end{cases} \]
1D Linear Hierarchical Basis

Figure: Nodal (left) and hierarchical basis (right) of level 3
2D Linear Hierarchical Basis

$$a^i_k(\xi) = a^i_{k_1} \otimes \cdots \otimes a^i_{k_N} = \prod_{j=1}^{j=N} a^i_{k_j}, \quad k = (k_1, \ldots, k_N)$$

**Figure:** 2D hierarchical basis $a^i_k = a^i_2 \otimes a^i_1$, $k = (1, 1)$ and $i = (2, 1)$
Constructing a 2D Sparse Grid

Figure: Support of the basis functions of the hierarchical subspaces
Recall

\[ \Delta^i(u) = U^i(u) - U^{i-1}(u) \]

and of course

\[ U^{i-1}(u) = U^i(U^{i-1}(u)) \]

So

\[ \Delta^i(u) = \sum_{\xi_k \in X^i} u(\xi_k) \cdot a^i_k - \sum_{\xi_k \in X^i} U^{i-1}(u)(\xi_k) \cdot a^i_k \]

\[ = \sum_{\xi_k \in X^i} \cdot a^i_k (u(\xi_k) - U^{i-1}(u)(\xi_k)) \]
From a Nodal Basis to A Hierarchical Basis

Noting that

$$u(\xi^i_k) - \mathcal{U}^{i-1}(u)(\xi^i_k) = 0 \quad \forall \xi^i_k \in X^{i-1}$$

Then

$$\Delta^i(u) = \sum_{\xi^i_k \in X^i} a_k^i \cdot (u(\xi^i_k) - \mathcal{U}^{i-1}(u)(\xi^i_k))$$

where $X^i_\Delta = X^i \setminus X^{i-1}$ and $m^i_\Delta = m^i - m^{i-1}$
1D Hierarchical Surpluses

In general

\[ \Delta^i(u) = \sum_{k=1}^{m_i^\Delta} a_k^i \left( u(\xi_k^i) - \mathcal{U}^{i-1}(u)(\xi_k^i) \right) \]

For linear hierarchical basis

\[ w_k^i = u(\xi_k^i) - \frac{u(\xi_k^i - h) + u(\xi_k^i + h)}{2} \]

where \( h = 1/(m^i - 1) \)
1D Linear Hierarchical Basis

Figure: Interpolation using the hierarchical basis
Multi-Dimensional Hierarchical Surpluses

In general

\[ w_k^i = u(\xi_{k1}^{i1}, \ldots, \xi_{kN}^{iN}) - A_{q-1,N}(u)(\xi_{k1}^{i1}, \ldots, \xi_{kN}^{iN}) \]

For linear hierarchical basis

\[ w_k^i = \left( \prod_{j=1}^{N} \left[ \begin{array}{ccc} -\frac{1}{2} & 1 & -\frac{1}{2} \\ \end{array} \right]_{i_j,k_j} \right) u \]

**Figure:** Stencil used to calculate linear hierarchical surpluses in 2D
Evaluating the Sparse Grid Interpolant

We can approximate any function \( u \in L_2(\Omega, \mathcal{F}, \mathcal{P}) \) by

\[
  u(x, t, \xi) = \sum_{|i| \leq q} \sum_{k \in B^i} w^i_k(x, t) \cdot a^i_k(\xi)
\]

using a new multi-index set to ease representation

\[
  B^i = \left\{ k \in \mathbb{N}^N \mid \xi_{kj}^i \in X_{\Delta}^{ij} \text{ for } k_j = 1, \ldots, m_{\Delta}^{ij}, j = 1, \ldots, N \right\}
\]
Locally Adaptive Sparse Grids

1. Construct first level sparse grid
Locally Adaptive Sparse Grids

1. Construct first level sparse grid
2. Evaluate function at the points in the first level grid.
Locally Adaptive Sparse Grids

1. Construct first level sparse grid
2. Evaluate function at the points in the first level grid.
3. Generate $2N$ Neighbour points of each point in first level grid and add each point to the active index set
Locally Adaptive Sparse Grids

Figure: The Initial adaptive Sparse Grid present in all adaptive Sparse Grids
Locally Adaptive Sparse Grids

1. Construct first level sparse grid
2. Evaluate function at the points in the first level grid.
3. Generate $2N$ neighbour points of each point in first level grid and add each point to the active index set.
4. Copy active index set to old index set and clear active index set.
Locally Adaptive Sparse Grids

1. Construct first level sparse grid
2. Evaluate function at the points in the first level grid.
3. Generate $2N$ Neighbour points of each point in first level grid and add each point to the active index set.
4. Copy active index set to old index set and clear active index set.
5. Calculate hierarchical surplus at each point in the old index set.
Locally Adaptive Sparse Grids

1. Construct first level sparse grid
2. Evaluate function at the points in the first level grid.
3. Generate $2^N$ Neighbour points of each point in first level grid and add each point to the active index set.
4. Copy active index set to old index set and clear active index set.
5. Calculate hierarchical surplus at each point in the old index set.
6. For each point with $|w_k^i| \geq \text{tol}$ generate $2^N$ Neighbour points and add to active set.
Locally Adaptive Sparse Grids

Figure: An example of adaptive refinement
Locally Adaptive Sparse Grids

1. Construct first level sparse grid
2. Evaluate function at the points in the first level grid.
3. Generate $2N$ Neighbour points of each point in first level grid and add each point to the active index set.
4. Copy active index set to old index set and clear active index set.
5. Calculate hierarchical surplus at each point in the old index set.
6. For each point with $|w_k^i| \geq \text{tol}$ Generate $2N$ Neighbour points and add to active set.
7. Add points in the old index set to the adaptive sparse grid.
Locally Adaptive Sparse Grids

1. Construct first level sparse grid
2. Evaluate function at the points in the first level grid.
3. Generate $2N$ neighbour points of each point in first level grid and add each point to the active index set.
4. Copy active index set to old index set and clear active index set.
5. Calculate hierarchical surplus at each point in the old index set.
6. For each point with $|w_k^i| \geq tol$ Generate $2N$ neighbour points and add to active set.
7. Add points in the old index set to the adaptive sparse grid.
8. Repeat steps 4-7.
Sparse grid approximation of a discontinuous function

**Figure:** Plot of \( f(x, y) = \frac{1}{|0.3 - x^2 - y^2| = \delta} \) on \([0, 1]^2\) where \(\delta = 0.1\)
Sparse grid approximation of a discontinuous function

Figure: Support nodes (5702) of the adaptive sparse grid with $\text{tol}=0.01$
Sparse grid approximation of a discontinuous function

**Figure:** Support nodes (720897) of the isotropic sparse grid
Calculating Important Statistics

The mean of the solution can be evaluated using

$$\mathbb{E}[u(x, t, \xi)] = \sum_{|i| \leq q} \sum_{|k| \in B^i} w^i_k(x, t) \cdot \int_{\Gamma} a^i_k(\xi) d\xi$$

where

$$\int_0^1 a^i_k(\xi) d\xi = \begin{cases} 1 & \text{if } i = 1 \\ \frac{1}{4} & \text{if } i = 2 \\ 2^{1-i} & \text{otherwise} \end{cases}$$

can be calculated analytically.
Consider the Kraichnan-Orszag three-mode Problem

\[
\begin{align*}
\frac{dy_1}{dt} &= y_1 y_3 \\
\frac{dy_2}{dt} &= -y_2 y_3 \\
\frac{dy_3}{dt} &= -y_1^2 + y_2^2
\end{align*}
\]

subject to initial conditions

\[
y_1(0; \omega) = \xi^1 \quad y_2(0; \omega) = \xi^2 \quad y_3(0; \omega) = \xi^3
\]
An example: 1D Kraichnan-Orszag Problem

Figure: Time evolution of the solutions $y = (y_1, y_2, y_3)$ to the orszag problem with 1D input $y_1(0) = 1.0$, $y_2(0; \omega) = 0.1\xi^1$, $y_3(0) = 0$
Figure: Evolution of the mean of the solution to the orszag problem with 1D input $y_1(0) = 1.0$, $y_2(0; \omega) = 0.1 \xi^1$, $y_3(0) = 0$. The number of grid points = 789.
**Figure:** 100 term gPC approximation of the solution to the 1D Kraichnan-Orszag Problem
An example: 2D Kraichnan-Orszag Problem

Figure: Evolution of the mean of the solution $y_1$ to the orszag problem with 2D input $y_1(0) = 1.0$, $y_2(0; \omega) = 0.1\xi^1$, $y_3(0; \omega) = \xi^2$. The number of grid points = 13595.