Bayes Linear Methods for Multiscale Emulation of a Hydrocarbon Reservoir

Jonathan Cumming

1st October 2008
Bayes Linear Methods for Multiscale Emulation of a Hydrocarbon Reservoir (and a bit of design too)

Jonathan Cumming

1st October 2008
Uncertainty in Computer Models

The (Very) General Problem

- Uncertainty analysis for a complex physical system given a model for that system

Jonathan Cumming

Bayes Linear Methods for Multiscale Emulation of a Hydrocarbon Reservoir (and a bit of design too)
The (Very) General Problem

- Uncertainty analysis for a complex physical system given a model for that system
- Complex physical system – infeasible to investigate the system, too complex to analyse exactly
The (Very) General Problem

- Uncertainty analysis for a complex physical system given a model for that system
- Complex physical system – infeasible to investigate the system, too complex to analyse exactly
- Computer model – construct a computer simulation which implements the known underlying mathematical and physical equations and laws which govern the behaviour of the system given some collection of input parameters
The (Very) General Problem

- **Uncertainty analysis** for a complex physical system given a model for that system
- Complex physical system – infeasible to investigate the system, too complex to analyse exactly
- Computer model – construct a computer simulation which implements the known underlying mathematical and physical equations and laws which govern the behaviour of the system given some collection of input parameters
- **Uncertainty analysis** – there are substantial uncertainties surrounding all key elements of the problem
Uncertainty in Computer Models

The (Very) General Problem

- Uncertainty analysis for a complex physical system given a model for that system
- Complex physical system – infeasible to investigate the system, too complex to analyse exactly
- Computer model – construct a computer simulation which implements the known underlying mathematical and physical equations and laws which govern the behaviour of the system given some collection of input parameters
- Uncertainty analysis – there are substantial uncertainties surrounding all key elements of the problem
- We want to learn and make inference about the system, its behaviour and its inputs

Jonathan Cumming
Bayes Linear Methods for Multiscale Emulation of a Hydrocarbon Reservoir (and a bit of design too)
Key features

- Key features of this class of problems:
  - The system is sufficiently complex
  - The model is expensive to evaluate
  - The model is informative for the system
  - The model is deterministic
  - (Some of) The inputs affect the model output (and hence the system)
Sources of Uncertainty

- Terminology: the physical system $y$, the model for that system $F(\cdot)$, and a vector of model inputs $x$
Sources of Uncertainty

- Terminology: the physical system $y$, the model for that system $F(\cdot)$, and a vector of model inputs $x$
- There are four major sources of uncertainty in a typical computer model problem:
Uncertainty in Computer Models

Sources of Uncertainty

- Terminology: the physical system $\mathbf{y}$, the model for that system $F(\cdot)$, and a vector of model inputs $\mathbf{x}$

- There are four major sources of uncertainty in a typical computer model problem:
  - **Model uncertainty** – we do not know the value of the model $F(\cdot)$ until we evaluate it at a given $\mathbf{x}$, but running the model is expensive
Model uncertainty

Jonathan Cumming
Bayes Linear Methods for Multiscale Emulation of a Hydrocarbon Reservoir (and a bit of design too)
Model uncertainty

Jonathan Cumming
Bayes Linear Methods for Multiscale Emulation of a Hydrocarbon Reservoir (and a bit of design too)
Uncertainty in Computer Models

### Sources of Uncertainty

- **Terminology:** the physical system $\mathbf{y}$, the model for that system $F(\cdot)$, and a vector of model inputs $\mathbf{x}$
- **There are four major sources of uncertainty in a typical computer model problem:**
  - **Model uncertainty** – we do not know the value of the model $F(\cdot)$ until we evaluate it at a given $\mathbf{x}$, but running the model is expensive.
  - **Input uncertainty** – we do not know what the ‘inputs’ to the physical system are, or what the corresponding input to the computer model should be.
Input uncertainty

Jonathan Cumming
Bayes Linear Methods for Multiscale Emulation of a Hydrocarbon Reservoir (and a bit of design too)
Sources of Uncertainty

- Terminology: the physical system $y$, the model for that system $F(\cdot)$, and a vector of model inputs $x$

- There are four major sources of uncertainty in a typical computer model problem:
  - Model uncertainty – we do not know the value of the model $F(\cdot)$ until we evaluate it at a given $x$, but running the model is expensive
  - Input uncertainty – we do not know what the ‘inputs’ to the physical system are, or what the corresponding input to the computer model should be
  - Model discrepancy – the model is wrong, $F(\cdot)$ is an imperfect description of $y$ and since $y$ is unknown we don’t know the extent of this inaccuracy

Jonathan Cumming
Bayes Linear Methods for Multiscale Emulation of a Hydrocarbon Reservoir (and a bit of design too)
Model discrepancy

Jonathan Cumming
Bayes Linear Methods for Multiscale Emulation of a Hydrocarbon Reservoir (and a bit of design too)
Uncertainty in Computer Models

**Sources of Uncertainty**

- Terminology: the physical system $y$, the model for that system $F(\cdot)$, and a vector of model inputs $x$

- There are four major sources of uncertainty in a typical computer model problem:
  - **Model uncertainty** – we do not know the value of the model $F(\cdot)$ until we evaluate it at a given $x$, but running the model is expensive
  - **Input uncertainty** – we do not know what the ‘inputs’ to the physical system are, or what the corresponding input to the computer model should be
  - **Model discrepancy** – the model is wrong, $F(\cdot)$ is an imperfect description of $y$ and since $y$ is unknown we don’t know the extent of this inaccuracy
  - **Observational error** – we don’t know the true value of the system $y$, we only observe noisy realisations $z$
Observational error

Jonathan Cumming
Bayes Linear Methods for Multiscale Emulation of a Hydrocarbon Reservoir (and a bit of design too)
Bayes Linear v. Probabilistic Bayes

- Differ fundamentally in the way they characterise uncertainty
Bayes Linear v. Probabilistic Bayes

- Differ fundamentally in the way they characterise uncertainty
- ‘Full’ Bayes – uncertainty is described by probability, all uncertain quantities have associated pdfs
- Bayes Linear – uncertainty is described by expectation, uncertain quantities described by $E[X]$ and $Var[X]$
Bayes Linear v. Probabilistic Bayes

- Differ fundamentally in the way they characterise uncertainty
- ‘Full’ Bayes – uncertainty is described by probability, all uncertain quantities have associated pdfs
- Bayes Linear – uncertainty is described by expectation, uncertain quantities described by $E[X]$ and $\text{Var}[X]$
- Advantages of probabilistic methods:
  - Suitable for problems of small or moderate size
  - Based in probability, can answer probabilistic questions
  - Reasonable approach if a meaningful elicitation is possible
  - Well-established
Bayes Linear v. Probabilistic Bayes

- Differ fundamentally in the way they characterise uncertainty
- ‘Full’ Bayes – uncertainty is described by probability, all uncertain quantities have associated pdfs
- Bayes Linear – uncertainty is described by expectation, uncertain quantities described by $E[X]$ and $\text{Var}[X]$
- **So why Bayes Linear then?**
  - Suitable for problems of small or moderate size
  - Based in probability, can answer probabilistic questions
  - Reasonable approach if a meaningful elicitation is possible
  - Well-established
Bayes Linear v. Probabilistic Bayes

- Differ fundamentally in the way they characterise uncertainty
- ‘Full’ Bayes – uncertainty is described by probability, all uncertain quantities have associated pdfs
- Bayes Linear – uncertainty is described by expectation, uncertain quantities described by $E[X]$ and $\text{Var}[X]$
- So why Bayes Linear then?
  - Computations are difficult, complex and intensive for larger problems
  - Based in probability, can answer probabilistic questions
  - Reasonable approach if a meaningful elicitation is possible
  - Well-established
Bayes Linear v. Probabilistic Bayes

- Differ fundamentally in the way they characterise uncertainty
- ‘Full’ Bayes – uncertainty is described by probability, all uncertain quantities have associated pdfs
- Bayes Linear – uncertainty is described by expectation, uncertain quantities described by $E[X]$ and $\text{Var}[X]$
- So why Bayes Linear then?
  - Computations are difficult, complex and intensive for larger problems
  - Expectation as primitive; likelihood functions become very complicated and so method can be very non-robust
  - Reasonable approach if a meaningful elicitation is possible
  - Well-established
Bayes Linear v. Probabilistic Bayes

- Differ fundamentally in the way they characterise uncertainty
- ‘Full’ Bayes – uncertainty is described by probability, all uncertain quantities have associated pdfs
- Bayes Linear – uncertainty is described by expectation, uncertain quantities described by $\mathbb{E}[X]$ and $\text{Var}[X]$
- So why Bayes Linear then?
  - Computations are difficult, complex and intensive for larger problems
  - Expectation as primitive; likelihood functions become very complicated and so method can be very non-robust
  - Extremely difficult to perform meaningful elicitation over high-dimensional spaces
  - Well-established
Bayes Linear v. Probabilistic Bayes

- Differ fundamentally in the way they characterise uncertainty
- ‘Full’ Bayes – uncertainty is described by probability, all uncertain quantities have associated pdfs
- Bayes Linear – uncertainty is described by expectation, uncertain quantities described by $E[X]$ and $\text{Var}[X]$
- So why Bayes Linear then?
  - Computations are difficult, complex and intensive for larger problems
  - Expectation as primitive; likelihood functions become very complicated and so method can be very non-robust
  - Extremely difficult to perform meaningful elicitation over high-dimensional spaces
  - Just too complex for large problems
Example: A Hydrocarbon Reservoir

- A oil reservoir is an underground region of porous rock which contains and oil and/or gas
- The hydrocarbons are trapped above by a layer of impermeable rock and below by a body of water, thus creating the reservoir
Example: A Hydrocarbon Reservoir

- A oil reservoir is an underground region of porous rock which contains and oil and/or gas
- The hydrocarbons are trapped above by a layer of impermeable rock and below by a body of water, thus creating the reservoir
- The oil and gas are pumped out of the reservoir (for £££), and sometimes fluids is pumped into the reservoir (to boost production)
- The purpose of the simulator is to model the flows and distributions of contents of the reservoir over time

Jonathan Cumming
Bayes Linear Methods for Multiscale Emulation of a Hydrocarbon Reservoir (and a bit of design too)
Example: A Hydrocarbon Reservoir

Inputs

- Each cell in the reservoir has a collection of associated parameters
  - Permeability, porosity
- There are also several other parameters
  - Fault transmissibility, Aquifer features, Saturation properties
- Since there are a huge number of these cells in the reservoir it is common to use a scalar multiplier over the whole reservoir (or subregions) to adjust values
Example: A Hydrocarbon Reservoir

Outputs

- The model outputs comprise the behaviour of the various wells and injectors in the reservoir.
- Output is a time series on the following variables for each well:
  - *Pressures*  Bottom-hole pressure, Tubing head pressure
  - *Production/Injection rates and totals* for each of oil, water and gas.
  - *Fluid ratios*  Water cut, Gas-oil ratio
- The resolution of the time series can be varied from months to years.
- With a large number of wells, daily output, or a long operating period there will be a *lot* of output data.
Gullfaks

- Gullfaks is a Norwegian hydrocarbon reservoir located in the North Sea
- The model is based on a grid of size $38 \times 87 \times 25$ and contains 43 production and 13 injection wells
- The model simulates 10 years of production taking 1.5–3 hours per simulation
- Inputs: Field multipliers for porosity ($\phi$), permeabilities ($k_x, k_z$), critical saturation ($crw$), and aquifer properties ($A_p, A_h$)
- Outputs: Focus on oil production rate for a 3-year period

Jonathan Cumming
Bayes Linear Methods for Multiscale Emulation of a Hydrocarbon Reservoir (and a bit of design too)
Example: A Hydrocarbon Reservoir

Map

Jonathan Cumming

Bayes Linear Methods for Multiscale Emulation of a Hydrocarbon Reservoir (and a bit of design too)
Bayes Linear Methods for Multiscale Emulation of a Hydrocarbon Reservoir (and a bit of design too)
Example: A Hydrocarbon Reservoir

Preliminaries

- Obtain a coarse simulator $F^c(x)$ by coarsening the vertical gridding by a factor of 10
- 1000 runs of $F^c(x)$ in a LHC over the input parameters
Example: A Hydrocarbon Reservoir

Preliminaries

- Obtain a coarse simulator $F_c(x)$ by coarsening the vertical gridding by a factor of 10
- 1000 runs of $F_c(x)$ in a LHC over the input parameters
- Focus on a 3-year window in the outputs, and the 10 production wells active in that period
- Consider the output quantity ‘oil production rate’ at each well
- Take 4-month averages over the time series
Preliminaries

- Obtain a coarse simulator $F^c(x)$ by coarsening the vertical gridding by a factor of 10
- 1000 runs of $F^c(x)$ in a LHC over the input parameters
- Focus on a 3-year window in the outputs, and the 10 production wells active in that period
- Consider the output quantity ‘oil production rate’ at each well
- Take 4-month averages over the time series
- Screen the wells (PV methods) – 4 wells capture 87% of the variation of the collection
Putting it all together

- We have a system $y$, a model $F(x)$, and some observations on the system, $z$.
- Link these quantities by two key equations:

$$ z = y + e $$

(1)
Putting it all together

- We have a system \( y \), a model \( F(x) \), and some observations on the system, \( z \).
- Link these quantities by two key equations:

\[
\begin{align*}
  z &= y + e \\
  y &= F(x^*) + \epsilon(x^*)
\end{align*}
\]
Putting it all together

- We have a system $\mathbf{y}$, a model $F(\mathbf{x})$, and some observations on the system, $\mathbf{z}$.
- Link these quantities by two key equations:
  \begin{align*}
  \mathbf{z} &= \mathbf{y} + \mathbf{e} \\
  \mathbf{y} &= F(\mathbf{x}^*) + \mathbf{\epsilon}(\mathbf{x}^*)
  \end{align*}

- Since complex computer models are expensive to evaluate we cannot run them at every parameter combination of interest.
Putting it all together

- We have a system $y$, a model $F(x)$, and some observations on the system, $z$.
- Link these quantities by two key equations:
  
  $$z = y + e$$
  
  $$y = F(x^*) + \epsilon(x^*)$$

- Since complex computer models are expensive to evaluate we cannot run them at every parameter combination of interest.
- Information is therefore limited about the computer simulation creating our “model uncertainty”
Putting it all together

- We have a system $y$, a model $F(x)$, and some observations on the system, $z$.
- Link these quantities by two key equations:
  \[
  z = y + e \\
  y = F(x^*) + \epsilon(x^*)
  \]
- Since complex computer models are expensive to evaluate we cannot run them at every parameter combination of interest.
- Information is therefore limited about the computer simulation creating our “model uncertainty”
- To address this model uncertainty, we construct a stochastic representation of the computer model – the emulator
The Emulator

- The emulator (or meta-model) is a statistical model of the deterministic computer model of the system:

Jonathan Cumming
Bayes Linear Methods for Multiscale Emulation of a Hydrocarbon Reservoir (and a bit of design too)
The Emulator

The emulator (or meta-model) is a statistical model of the deterministic computer model of the system:

\[ f_i(x) = g_i(x_{[i]})^T \beta_i + u_i(x_{[i]}) + v_i(x), \tag{3} \]
The Emulator

- The emulator (or meta-model) is a statistical model of the deterministic computer model of the system:

\[ f_i(x) = g_i(x_{[i]})^T \beta_i + u_i(x_{[i]}) + v_i(x), \]

- \( g_i(x_{[i]})^T \beta_i \) – a global trend function which captures the gross features of the model
The Emulator

- The emulator (or meta-model) is a statistical model of the deterministic computer model of the system:

\[ f_i(x) = g_i(x[i])^T \beta_i + u_i(x[i]) + v_i(x), \]  

(3)

- \( g_i(x[i])^T \beta_i \) – a global trend function which captures the gross features of the model

- \( x[i] \) – a subset of inputs which account for most of the variation in \( F \), the active variables
The Emulator

- The emulator (or meta-model) is a statistical model of the deterministic computer model of the system:

\[ f_i(x) = g_i(x[i])^T \beta_i + u_i(x[i]) + v_i(x), \] (3)

- \( g_i(x[i])^T \beta_i \) – a global trend function which captures the gross features of the model
- \( x[i] \) – a subset of inputs which account for most of the variation in \( F \), the *active variables*
- \( u_i(x[i]) + v_i(x) \) – a stochastic component representing the local behaviour in the model residuals
The Emulator

- The emulator (or meta-model) is a statistical model of the deterministic computer model of the system:

\[
    f_i(x) = g_i(x[i])^T \beta_i + u_i(x[i]) + v_i(x),
\]

- \( g_i(x[i])^T \beta_i \) – a global trend function which captures the gross features of the model
- \( x[i] \) – a subset of inputs which account for most of the variation in \( F \), the active variables
- \( u_i(x[i]) + v_i(x) \) – a stochastic component representing the local behaviour in the model residuals
- \( u_i(x[i]) \) – a correlated residual process in the AVs
The Emulator

- The emulator (or meta-model) is a statistical model of the deterministic computer model of the system:

\[ f_i(x) = g_i(x_{[i]})^T \beta_i + u_i(x_{[i]}) + v_i(x), \]  

(3)

- \( g_i(x_{[i]})^T \beta_i \) – a global trend function which captures the gross features of the model

- \( x_{[i]} \) – a subset of inputs which account for most of the variation in \( F \), the active variables

- \( u_i(x_{[i]}) + v_i(x) \) – a stochastic component representing the local behaviour in the model residuals

- \( u_i(x_{[i]}) \) – a correlated residual process in the AVs

- \( v_i(x) \) – an uncorrelated ‘nugget’ residual
Emulator Components

- Model trend – represents the large-scale model effects over the whole input space, very useful when coverage is sparse
Single-Simulator Emulation

**Emulator Components**

- Model trend – represents the large-scale model effects over the whole input space, very useful when coverage is sparse

- Active variables $x_{[i]}$ – often most of the model variation is driven by a few key inputs, and the others have minimal contribution. Making this partition reduces the complexity of the calculations and subsequent analyses
Emulator Components

• Model trend – represents the large-scale model effects over the whole input space, very useful when coverage is sparse
• Active variables \( \mathbf{x}_{[i]} \) – often most of the model variation is driven by a few key inputs, and the others have minimal contribution. Making this partition reduces the complexity of the calculations and subsequent analyses
• Correlated Residual \( u_i(\mathbf{x}_{[i]}) \) – a correlated stochastic process, which accounts for any additional variation in model due to active variables that is not picked up by trend.
• It is correlated since nearby points in the input space will likely have similar values
Emulator Components contd.

- We use a correlation function to describe the spatial residual correlation, such as the Gaussian form

\[
\text{Cov} \left[ u_i^c(x_i), u_i^c(x_i') \right] = \sigma^2_{u_i} \exp \left[ - \sum_j \left( \frac{x_i[j] - x_i'[j]}{\theta_{ij}} \right)^2 \right]
\]
Emulator Components contd.

- We use a correlation function to describe the spatial residual correlation, such as the Gaussian form

\[
\text{Cov} \left[ u_i^c(x[i]), u_i^c(x'[i]) \right] = \sigma_{u_i}^2 \exp \left[ - \sum_j \left( \frac{x[i]_j - x'[i]_j}{\theta_{ij}} \right)^2 \right]
\]

- Nugget \( v_i(x) \) – an unstructured residual process, which soaks up any and all remaining variation not captured by the trend or the residual process. The contribution of the nugget process is small, and so treating it as uncorrelated has negligible impact on the calculations.
Coarse and Accurate Emulators

- The computer model is expensive to evaluate, so we have a limited amount of information about its behaviour
Coarse and Accurate Emulators

- The computer model is expensive to evaluate, so we have a limited amount of information about its behaviour.
- In some cases, approximate versions of the same computer model are available which could be used to help learn about the expensive model.
Coarse and Accurate Emulators

- The computer model is expensive to evaluate, so we have a limited amount of information about its behaviour.
- In some cases, approximate versions of the same computer model are available which could be used to help learn about the expensive model.
- The ‘coarse’ model, $F^c(x)$, will capture many of the qualitative features of the full simulator, but be substantially cheaper allowing a large number of runs.
Coarse and Accurate Emulators

- The computer model is expensive to evaluate, so we have a limited amount of information about its behaviour.
- In some cases, approximate versions of the same computer model are available which could be used to help learn about the expensive model.
- The ‘coarse’ model, $F^c(\mathbf{x})$, will capture many of the qualitative features of the full simulator, but be substantially cheaper allowing a large number of runs.
- We can exploit the cheapness of the coarse model to perform many runs and emulate it well, construct a framework linking the two simulator and combine with a small number of runs of the full model to obtain an emulator of the expensive simulator.
Emulator Construction for the Coarse Simulator

- *n* is large so we fit emulators to each output individually by using information from the model runs.
Bayes Linear Methods for Multiscale Emulation

### Emulator Construction for the Coarse Simulator

- $n$ is large so we fit emulators to each output individually by using information from the model runs.
- Determine the active variables $x[i]$ and basis functions $g_i(\cdot)$ by model selection.
- Specify $E[\beta_i^c] = \hat{\beta}_i^c$, $\text{Var}[\beta_i^c] = \hat{\Sigma}_i^c$ where $\hat{\beta}_i^c$ and $\hat{\Sigma}_i^c$ are OLS estimates.
Emulator Construction for the Coarse Simulator

- $n$ is large so we fit emulators to each output individually by using information from the model runs.
- Determine the active variables $x_{[i]}$ and basis functions $g_i(\cdot)$ by model selection.
- Specify $\mathbb{E} [\beta_i^c] = \hat{\beta}_i^c$, $\text{Var} [\beta_i^c] = \hat{\Sigma}_i^c$ where $\hat{\beta}_i^c$ and $\hat{\Sigma}_i^c$ are OLS estimates.
- $u_i^c(x_{[i]})$ is a residual process with a Gaussian correlation function with correlation length $\theta$, $v_i^c(x)$ is a nugget process with zero correlation length.
- Consider $\text{Var} [w_i^c] = \hat{\sigma}^2$, $\text{Var} [u_i^c(x_{[i]})] = (1 - \delta)\hat{\sigma}^2$, $\text{Var} [v_i^c(x_{[i]})] = \delta\hat{\sigma}^2$.
- Obtain hyperparameter values by variogram methods.
Bayes Linear Methods for Multiscale Emulation of a Hydrocarbon Reservoir (and a bit of design too)
## Emulation Summaries

<table>
<thead>
<tr>
<th>Well</th>
<th>Time</th>
<th>$x_{[i]}$</th>
<th>No. Model Terms</th>
<th>Coarse Simulator $R^2$</th>
<th>Accurate Simulator $\tilde{R}^2$</th>
</tr>
</thead>
<tbody>
<tr>
<td>B2</td>
<td>4</td>
<td>$\phi, crw, A_p$</td>
<td>9</td>
<td>0.886</td>
<td>0.951</td>
</tr>
<tr>
<td>B2</td>
<td>8</td>
<td>$\phi, crw, A_p$</td>
<td>7</td>
<td>0.959</td>
<td>0.958</td>
</tr>
<tr>
<td>B2</td>
<td>12</td>
<td>$\phi, crw, A_p$</td>
<td>10</td>
<td>0.978</td>
<td>0.995</td>
</tr>
<tr>
<td>B2</td>
<td>16</td>
<td>$\phi, crw, k_z$</td>
<td>7</td>
<td>0.970</td>
<td>0.995</td>
</tr>
<tr>
<td>B2</td>
<td>20</td>
<td>$\phi, crw, k_x$</td>
<td>11</td>
<td>0.967</td>
<td>0.986</td>
</tr>
<tr>
<td>B2</td>
<td>24</td>
<td>$\phi, crw, k_x$</td>
<td>10</td>
<td>0.970</td>
<td>0.970</td>
</tr>
<tr>
<td>B2</td>
<td>28</td>
<td>$\phi, crw, k_x$</td>
<td>10</td>
<td>0.975</td>
<td>0.981</td>
</tr>
<tr>
<td>B2</td>
<td>32</td>
<td>$\phi, crw, k_x$</td>
<td>11</td>
<td>0.980</td>
<td>0.951</td>
</tr>
<tr>
<td>B2</td>
<td>36</td>
<td>$\phi, crw, k_x$</td>
<td>11</td>
<td>0.983</td>
<td>0.967</td>
</tr>
</tbody>
</table>
Linking the Emulators

- $F^c$ is cheap to evaluate so we generate a large space-filling design to get reasonable coverage of the input space.
Linking the Emulators

- $F^c$ is cheap to evaluate so we generate a large space-filling design to get reasonable coverage of the input space.
- Since $n$ is large, we fit emulators to each output individually and using information from the model runs to get $f^c_i(x)$. 

Jonathan Cumming
Bayes Linear Methods for Multiscale Emulation of a Hydrocarbon Reservoir (and a bit of design too)
Bayes Linear Methods for Multiscale Emulation

Linking the Emulators

- $F^c$ is cheap to evaluate so we generate a large space-filling design to get reasonable coverage of the input space.
- Since $n$ is large, we fit emulators to each output individually and using information from the model runs to get $f_i^c(x)$.
- We consider the emulators to have the form

$$f_i^c(x) = g_i(x[i])^T \beta_i^c + w_i^c(x), \quad (4)$$

$$f_i^a(x) = g_i(x[i])^T \beta_i^a + \beta_{wi}^c w_i^c(x) + w_i^a(x), \quad (5)$$
Bayes Linear Methods for Multiscale Emulation

Linking the Emulators

- $F^c$ is cheap to evaluate so we generate a large space-filling design to get reasonable coverage of the input space.
- Since $n$ is large, we fit emulators to each output individually and using information from the model runs to get $f^c_i(x)$.
- We consider the emulators to have the form

$$f^c_i(x) = g_i(x[i])^T \beta_i^c + \epsilon_i^c(x), \quad (4)$$

$$f^a_i(x) = g_i(x[i])^T \beta_i^a + \beta_{wi}^a \epsilon_i^c(x) + \epsilon_i^a(x), \quad (5)$$

- We have the same $g_i(\cdot)$ and $x[i]$ in $f^c$ and $f^a$.
Bayes Linear Methods for Multiscale Emulation of a Hydrocarbon Reservoir (and a bit of design too)

**Linking the Emulators**

- $F^c$ is cheap to evaluate so we generate a large space-filling design to get reasonable coverage of the input space.
- Since $n$ is large, we fit emulators to each output individually and using information from the model runs to get $f^c_i(x)$.
- We consider the emulators to have the form

$$f^c_i(x) = g_i(x_{[i]})^T \beta^c_i + w^c_i(x),$$

$$f^a_i(x) = g_i(x_{[i]})^T \beta^a_i + \beta^a_{wi} w^c_i(x) + w^a_i(x),$$

- We have the same $g_i(\cdot)$ and $x_{[i]}$ in $f^c$ and $f^a$.
- Link (4) and (5) via the coefficients

$$\beta^a_{ij} = \rho_{ij} \beta^c_{ij} + \gamma_{ij}$$

Jonathan Cumming
Other Methods

- Some alternative multiscale emulation methods directly embed the coarse simulator into the emulator for the accurate model.
Other Methods

- Some alternative multiscale emulation methods directly embed the coarse simulator into the emulator for the accurate model.
- Kennedy & O’Hagan (2000) have a single multiplier to control the effect of the coarse model on the fine:

\[ f_i^a(x) = \rho F^c(x) + w_i^a(x), \quad (7) \]
Other Methods

- Some alternative multiscale emulation methods directly embed the coarse simulator into the emulator for the accurate model.

- Kennedy & O’Hagan (2000) have a single multiplier to control the effect of the coarse model on the fine:

  \[ f_i^a(x) = \rho F^c(x) + w_i^a(x), \]  
  \[ (7) \]

- Qian & Wu (2008) generalise this and make the multiplier a stochastic process:

  \[ f_i^a(x) = \rho(x)F^c(x) + w_i^a(x) \]  
  \[ (8) \]
Bayes Linear Methods for Multiscale Emulation of a Hydrocarbon Reservoir (and a bit of design too)
Fine Model Runs

- Given the coarse emulators \( f_i^c(x) \) and specifications for the multilevel parameters \( \rho_{ij} \) and \( \gamma_{ij} \) we can construct a prior emulator for the accurate simulator.
- We now want to update this prior emulator by runs of the accurate model.
- Q: At what locations should we run the fine computer model?
Fine Model Runs

- Given the coarse emulators $f_i^c(x)$ and specifications for the multilevel parameters $\rho_{ij}$ and $\gamma_{ij}$ we can construct a prior emulator for the accurate simulator.
- We now want to update this prior emulator by runs of the accurate model.
- Q: At what locations should we run the fine computer model?
- We could design over the space of all model inputs...
Bayes Linear Methods for Multiscale Emulation

Fine Model Runs

- Given the coarse emulators $f_i^c(x)$ and specifications for the multilevel parameters $\rho_{ij}$ and $\gamma_{ij}$ we can construct a prior emulator for the accurate simulator.

- We now want to update this prior emulator by runs of the accurate model.

- Q: At what locations should we run the fine computer model?

- We could design over the space of all model inputs...

- ... but there is structural information here that we could use to produce a more informed design.
Standard approaches to design for computer experiments fall into two categories:
Design for Computer Models

- Standard approaches to design for computer experiments fall into two categories:
  - **Space-filling methods** – choose points which are reasonably evenly spread throughout the input space to obtain uniform coverage
  - Includes: Latin hypercubes, Low-discrepancy sequences (e.g. Sobol’)

Jonathan Cumming
Bayes Linear Methods for Multiscale Emulation of a Hydrocarbon Reservoir (and a bit of design too)
Designing for Multiscale Emulators

Design for Computer Models

- Standard approaches to design for computer experiments fall into two categories:
  - Space-filling methods – choose points which are reasonably evenly spread throughout the input space to obtain uniform coverage
  - Includes: Latin hypercubes, Low-discrepancy sequences (e.g Sobol’)
  - Criterion-based methods – choose points which optimise a particular criterion
  - Includes: maximum entropy, minimum MSPE/IMSPE
An Example

Example

Suppose a simulator $F^a(x)$ has two outputs. Suppose also that we have used an approximate version to construct coarse emulators $f_1^c(x)$ and $f_2^c(x)$. The inputs $x$ are 10-dimensional, i.e. $x = (x_1, \ldots, x_{10})$. 
An Example

Example

Suppose a simulator $F^a(x)$ has two outputs. Suppose also that we have used an approximate version to construct coarse emulators $f_1^c(x)$ and $f_2^c(x)$. The inputs $x$ are 10-dimensional, i.e. $x = (x_1, \ldots, x_{10})$.

- Naively, we would try to construct a space-filling design over the entire 10-dimensional input space.
An Example

Example

Suppose a simulator \( F^a(x) \) has two outputs. Suppose also that we have used an approximate version to construct coarse emulators \( f^c_1(x) \) and \( f^c_2(x) \). The inputs \( x \) are 10-dimensional, i.e. \( x = (x_1, \ldots, x_{10}) \).

- Naively, we would try to construct a space-filling design over the entire 10-dimensional input space.
- However, runs are expensive and so we will only be able to perform a small number.
An Example

Suppose a simulator $F^a(x)$ has two outputs. Suppose also that we have used an approximate version to construct coarse emulators $f_1^c(x)$ and $f_2^c(x)$. The inputs $x$ are 10-dimensional, i.e. $x = (x_1, \ldots, x_{10})$.

- Naively, we would try to construct a space-filling design over the entire 10-dimensional input space.
- However, runs are expensive and so we will only be able to perform a small number.
- Space-filling increasingly higher-dimensional spaces requires exponentially more runs (the curse of dimensionality).
An Example contd.

Example with Active Variables

Suppose we identify the AVs in the coarse emulators giving:

\[
f_1^c(x) = s_1^c(x_1, x_2, x_3, x_4) + w_1^c(x)
\]

\[
f_2^c(x) = s_1^c(x_1, x_2, x_5, x_6) + w_2^c(x)
\]
Example with Active Variables

Suppose we identify the AVs in the coarse emulators giving:

\[ f_1^c(x) = s_1^c(x_1, x_2, x_3, x_4) + w_1^c(x) \]
\[ f_2^c(x) = s_1^c(x_1, x_2, x_5, x_6) + w_2^c(x) \]

- By identifying active variables, we can see that not all inputs are active on this simulator.
Example with Active Variables

Suppose we identify the AVs in the coarse emulators giving:

\[ f_1^c(x) = s_1^c(x_1, x_2, x_3, x_4) + w_1^c(x) \]
\[ f_2^c(x) = s_1^c(x_1, x_2, x_5, x_6) + w_2^c(x) \]

- By identifying active variables, we can see that not all inputs are active on this simulator.
- AVs are \( \mathcal{X}^A = \{x_1...6\} \), inactive inputs are \( \mathcal{X}^U = \{x_7...10\} \).
Suppose we identify the AVs in the coarse emulators giving:

\[
\begin{align*}
    f_1^c(x) &= s_1^c(x_1, x_2, x_3, x_4) + w_1^c(x) \\
    f_2^c(x) &= s_1^c(x_1, x_2, x_5, x_6) + w_2^c(x)
\end{align*}
\]

- By identifying active variables, we can see that not all inputs are active on this simulator.
- AVs are \( \mathcal{X}^A = \{x_1...6\} \), inactive inputs are \( \mathcal{X}^U = \{x_7...10\} \).
- Only the AVs have ‘important’ effects on model output; concern ourselves only with designing for \( \mathcal{X}^A \).
An Example contd.

Example with Active Variables

Suppose we identify the AVs in the coarse emulators giving:

\[
\begin{align*}
  f_1^c(x) &= s_1^c(x_1, x_2, x_3, x_4) + w_1^c(x) \\
  f_2^c(x) &= s_1^c(x_1, x_2, x_5, x_6) + w_2^c(x)
\end{align*}
\]

- By identifying active variables, we can see that not all inputs are active on this simulator.
- AVs are \(\mathcal{X}^A = \{x_1...6\}\), inactive inputs are \(\mathcal{X}^U = \{x_7...10\}\).
- Only the AVs have ‘important’ effects on model output; concern ourselves only with designing for \(\mathcal{X}^A\).
- The problem is reduced from 10-D to 6-D giving better coverage of important inputs for a fixed design size.
Example with Active Variables

Emulators:

\[ f_c^1(x) = s_c^1(x_1, x_2, x_3, x_4) + w_c^1(x) \]
\[ f_c^2(x) = s_c^1(x_1, x_2, x_5, x_6) + w_c^2(x) \]

But \( \mathcal{X}_B^A = \{x_1, x_2\} \) are shared by both emulators, whereas \( \mathcal{X}_1^A = \{x_3, x_4\} \) and \( \mathcal{X}_2^A = \{x_5, x_6\} \) are unique to \( f_c^1 \) and \( f_c^2 \).
An Example contd.

<table>
<thead>
<tr>
<th>Example with Active Variables</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Emulators:</strong></td>
</tr>
<tr>
<td>$f_1^c(x) = s_1^c(x_1, x_2, x_3, x_4) + w_1^c(x)$</td>
</tr>
<tr>
<td>$f_2^c(x) = s_1^c(x_1, x_2, x_5, x_6) + w_2^c(x)$</td>
</tr>
</tbody>
</table>

- But $\mathcal{X}_A = \{x_1, x_2\}$ are shared by both emulators, whereas $\mathcal{X}_1^A = \{x_3, x_4\}$ and $\mathcal{X}_2^A = \{x_5, x_6\}$ are unique to $f_1^c$ and $f_2^c$.

- In general, $\mathcal{X}_B^A$ are the border variables and can interact freely with any other input.
Designing for Multiscale Emulators

An Example contd.

### Example with Active Variables

Emulators:

\[
\begin{align*}
    f_1^c(x) &= s_1^c(x_1, x_2, x_3, x_4) + w_1^c(x) \\
    f_2^c(x) &= s_1^c(x_1, x_2, x_5, x_6) + w_2^c(x)
\end{align*}
\]

- But \( \mathcal{X}_B^A = \{x_1, x_2\} \) are shared by both emulators, whereas \( \mathcal{X}_1^A = \{x_3, x_4\} \) and \( \mathcal{X}_2^A = \{x_5, x_6\} \) are unique to \( f_1^c \) and \( f_2^c \).
- In general, \( \mathcal{X}_B^A \) are the border variables and can interact freely with any other input.
- \( \mathcal{X}_i^A \) is the \( i \)th set of block variables, which can only interact with inputs within the same block or with inputs in \( \mathcal{X}_B^A \).
Active Variables and BBD matrices

- Given the border inputs, the block inputs only interact with other inputs within the same block and without any between block interaction.
Active Variables and BBD matrices

- Given the border inputs, the block inputs only interact with other inputs within the same block and without any between block interaction.
- Using this partition, given a design for $\mathcal{X}^A_B$ we can consider designs for the $\mathcal{X}^A_i$ separately.
Active Variables and BBD matrices

- Given the border inputs, the block inputs only interact with other inputs within the same block and without any between block interaction.
- Using this partition, given a design for $X_A$ we can consider designs for the $X^A_i$ separately.
- This structure has a relationship with graphs and sparse matrices.

![Diagram](attachment:image.png)
Design Construction

- The border and block define the collection of design problems that we must optimise over in choosing runs of the fine simulator.
Design Construction

- The border and block define the collection of design problems that we must optimise over in choosing runs of the fine simulator.
- The design problem breaks down into designing over a number of subsets of inputs, where some inputs are shared.
Design Construction

- The border and block define the collection of design problems that we must optimise over in choosing runs of the fine simulator.
- The design problem breaks down into designing over a number of subsets of inputs, where some inputs are shared.
- Since the border inputs interact both with one another and the blocks, we design for the border first.
Design Construction

- The border and block define the collection of design problems that we must optimise over in choosing runs of the fine simulator.
- The design problem breaks down into designing over a number of subsets of inputs, where some inputs are shared.
- Since the border inputs interact both with one another and the blocks, we design for the border first.
- Given the border design we can then ‘add-on’ designs for each of the blocks separately.
Design Construction

- The border and block define the collection of design problems that we must optimise over in choosing runs of the fine simulator.
- The design problem breaks down into designing over a number of subsets of inputs, where some inputs are shared.
- Since the border inputs interact both with one another and the blocks, we design for the border first.
- Given the border design we can then ‘add-on’ designs for each of the blocks separately.
- This requires that we can sensibly construct designs by augmenting with extra variables.
Design Construction

- The border and block define the collection of design problems that we must optimise over in choosing runs of the fine simulator.
- The design problem breaks down into designing over a number of subsets of inputs, where some inputs are shared.
- Since the border inputs interact both with one another and the blocks, we design for the border first.
- Given the border design we can then ‘add-on’ designs for each of the blocks separately.
- This requires that we can sensibly construct designs by augmenting with extra variables.
- Then we can iteratively refine the design until we are sufficiently satisfied with its performance.

Jonathan Cumming
Bayes Linear Methods for Multiscale Emulation of a Hydrocarbon Reservoir (and a bit of design too)
Design Criteria

- In order to compare two possible designs during the iterative design search, we require a design criterion
Design Criteria

- In order to compare two possible designs during the iterative design search, we require a design criterion.
- At this stage we seek a design which is effective in reducing uncertainty about $F^a(x)$. 

Jonathan Cumming
Bayes Linear Methods for Multiscale Emulation of a Hydrocarbon Reservoir (and a bit of design too)
In order to compare two possible designs during the iterative design search, we require a design criterion.

At this stage we seek a design which is effective in reducing uncertainty about $F^a(x)$.

This uncertainty is characterised by the adjusted variance of $f^a(x)$ given the model evaluations at the proposed design points $F^a$. 
Design Criteria

- In order to compare two possible designs during the iterative design search, we require a design criterion.
- At this stage we seek a design which is effective in reducing uncertainty about $F^a(x)$.
- This uncertainty is characterised by the adjusted variance of $f^a(x)$ given the model evaluations at the proposed design points $F^a$.
- Thus an appropriate design criterion would be

$$\begin{align*}
C &= \text{tr}\{\text{Var}_{F^a}[f^a(X)]\} \\
&\approx \text{tr}\{G_i^T\text{Var}_{F^a}[\beta_i^{a+}] G_i\} \\
&= \text{tr}\{\text{Var}_{F^a}[\beta_i^{a+}] G_i^*\}, \quad (9)
\end{align*}$$
Updating the Accurate Emulator

- Given the prior emulator for $F^a_i(x)$ and the simulator evaluations $F^a$, we now update our prior beliefs via the Bayes linear adjustment formulae.
Updating the Accurate Emulator

- Given the prior emulator for $F^a_i(x)$ and the simulator evaluations $F^a$, we now update our prior beliefs via the Bayes linear adjustment formulae.
- This gives the following adjusted expectation and variance:

$$E_{F^a_i}[f^a_i(x)] = E[f^a_i(x)] + \text{Cov}[f^a_i(x), F^a_i] \text{Var}[F^a_i]^{-1} [F^a_i - E[F^a_i]],$$

$$\text{Var}_{F^a_i}[f^a_i(x)] = \text{Var}[f^a_i(x)] - \text{Cov}[f^a_i(x), F^a_i] \text{Var}[F^a_i]^{-1} \text{Cov}[F^a_i, f^a_i(x)].$$
History Matching

- We want to reduce the size of the input space $\mathcal{X}$ by eliminating any input points whose simulator evaluations are unlikely to reproduce the behaviour of the physical system.
History Matching

- We want to reduce the size of the input space $\mathcal{X}$ by eliminating any input points whose simulator evaluations are unlikely to reproduce the behaviour of the physical system.
- Use the implausibility measure

$$I_{(i)}(x) = \frac{|\mathbb{E}[f_i^a(x)] - z_i|^2}{\text{Var}[f_i^a(x) - z_i]},$$

- Or the multivariate form

$$I(x) = (\mathbb{E}[f^a(x)] - z)^T \text{Var}[f^a(x) - z]^{-1} (\mathbb{E}[f^a(x)] - z)/q,$$
History Matching

- We want to reduce the size of the input space $\mathcal{X}$ by eliminating any input points whose simulator evaluations are unlikely to reproduce the behaviour of the physical system.
- Use the implausibility measure

$$I_{(i)}(x) = \frac{|\mathbb{E}[f^a_i(x)] - z_i|^2}{\text{Var}[f^a_i(x) - z_i]},$$

- Or the multivariate form

$$I(x) = (\mathbb{E}[f^a(x)] - z)^T \text{Var}[f^a(x) - z]^{-1} (\mathbb{E}[f^a(x)] - z)/q,$$

- Evaluate $I(x)$ over a grid in the input parameters and identify and exclude regions of poor match quality.
History Matching

- We want to reduce the size of the input space $\mathcal{X}$ by eliminating any input points whose simulator evaluations are unlikely to reproduce the behaviour of the physical system.
- Use the implausibility measure

$$I_{(i)}(x) = \frac{|E[f^a_i(x)] - z_i|^2}{\text{Var}[f^a_i(x) - z_i]},$$

- Or the multivariate form

$$I(x) = (E[f^a(x)] - z)^T \text{Var}[f^a(x) - z]^{-1} (E[f^a(x)] - z)/q,$$

- Evaluate $I(x)$ over a grid in the input parameters and identify and exclude regions of poor match quality.
- Obtain a reduced space $\mathcal{X}^*$ of potential matches.
Implausibility Results

Jonathan Cumming
Bayes Linear Methods for Multiscale Emulation of a Hydrocarbon Reservoir (and a bit of design too)
### Refocusing

- Make the restriction \( \mathcal{X}^* = \{ \mathbf{x} : \mathcal{I}(\mathbf{x}) \leq 4 \} \approx \{ \mathbf{x} : \phi < 0.79 \} \) and eliminate 90\% of the input space.
Refocusing

- Make the restriction $\mathcal{X}^* = \{x : \mathcal{I}(x) \leq 4\} \simeq \{x : \phi < 0.79\}$ and eliminate 90% of the input space
- Now consider final 4 time points in original data, plus an additional point 1 year beyond the end of the previous series to be forecast
Now what?

Refocusing

- Make the restriction $\mathcal{X}^* = \{ \mathbf{x} : I(\mathbf{x}) \leq 4 \} \simeq \{ \mathbf{x} : \phi < 0.79 \}$ and eliminate 90% of the input space.

- Now consider final 4 time points in original data, plus an additional point 1 year beyond the end of the previous series to be forecast.

- Since reducing the space many of the old model runs are no longer valid, so supplement with additional evaluations.

- 262+100 coarse runs, 6+20 accurate runs.

- Re-fit the coarse and fine emulators, using the old emulator structure as a starting point.
Forecasting

- We now forecast the “future” system value $y_P$ given the observed historical data $z_H$ (using our collection of emulators)

\[
E_{z_H} [y_P] = E [y_P] + \text{Cov} [y_P, z_H] \text{Var} [z_H]^{-1} (z_H - E [z_H])
\]

\[
\text{Var}_{z_H} [y_P] = \text{Var} [y_P] - \text{Cov} [y_P, z_H] \text{Var} [z_H]^{-1} \text{Cov} [z_H, y_P].
\]
Now what?

Forecasting

- We now forecast the “future” system value $y_P$ given the observed historical data $z_H$ (using our collection of emulators)

\[
E_{z_H} [y_P] = E [y_P] + \text{Cov} [y_P, z_H] \text{Var} [z_H]^{-1} (z_H - E [z_H])
\]

\[
\text{Var}_{z_H} [y_P] = \text{Var} [y_P] - \text{Cov} [y_P, z_H] \text{Var} [z_H]^{-1} \text{Cov} [z_H, y_P].
\]

- Plug in $z_H = y_H + e_H$ and $y = F^a(x^*) + \epsilon$
Forecasting

- We now forecast the “future” system value $y_P$ given the observed historical data $z_H$ (using our collection of emulators)

$$
E_{z_H} [y_P] = E [y_P] + \text{Cov} [y_P, z_H] \text{Var} [z_H]^{-1} (z_H - E [z_H])
$$

$$
\text{Var}_{z_H} [y_P] = \text{Var} [y_P] - \text{Cov} [y_P, z_H] \text{Var} [z_H]^{-1} \text{Cov} [z_H, y_P].
$$

- Plug in $z_H = y_H + e_H$ and $y = F^a(x^*) + \epsilon$

- Requires specification of model discrepancy $\epsilon$ and some prior specification for $x^*$
Now what?

Forecasting

- We now forecast the “future” system value $y_P$ given the observed historical data $z_H$ (using our collection of emulators)

\[
E_{z_H} [y_P] = E [y_P] + \text{Cov} [y_P, z_H] \text{Var} [z_H]^{-1} (z_H - E [z_H])
\]

\[
\text{Var}_{z_H} [y_P] = \text{Var} [y_P] - \text{Cov} [y_P, z_H] \text{Var} [z_H]^{-1} \text{Cov} [z_H, y_P].
\]

- Plug in $z_H = y_H + e_H$ and $y = F^a(x^*) + \epsilon$

- Requires specification of model discrepancy $\epsilon$ and some prior specification for $x^*$

\[
E_{z_H} [y_P] = \mu^*_P + (\Sigma^*_P + \Sigma^*_H + \Sigma^e_H)^{-1} (z_H - \mu^*_H)
\]

\[
\text{Var}_{z_H} [y_P] = (\Sigma^*_P + \Sigma^*_H)^{-1} (z_H - \mu^*_H)^T (\Sigma^*_P + \Sigma^*_H) (z_H - \mu^*_H)
\]
Forecasting Results

Jonathan Cumming
Bayes Linear Methods for Multiscale Emulation of a Hydrocarbon Reservoir (and a bit of design too)
Obligatory conclusion slide

- Application conclusions:
  - Of the inputs, only porosity ($\phi$) seemed to have any notable effect on model output
  - By history matching and re-focussing we have reduced the size of the feasible input space by 90%
  - Using Bayes Linear forecasting, we can obtain good forecasts for future production
  - Using the border-block decomposition substantially simplifies design construction
Obligatory conclusion slide

- Application conclusions:
  - Of the inputs, only porosity ($\phi$) seemed to have any notable effect on model output
  - By history matching and re-focussing we have reduced the size of the feasible input space by 90%
  - Using Bayes Linear forecasting, we can obtain good forecasts for future production
  - Using the border-block decomposition substantially simplifies design construction

- What’s next?
  - Generalise method to allow for more simulators
  - Diagnostics
  - Apply BBD factorisation to other calculations, eg implausibility