Kriging Metamodels for Bermudan Option Pricing

Mike Ludkovski

Dept of Statistics & Applied Probability UC Santa Barbara

USC Financial Math Seminar
September 21 2015
Work supported by NSF DMS-1222262
Bermudan Option Pricing/ Optimal Stopping

- State process $X$, payoff $h(t, X_t)$
- Discrete-time: $t = 1, 2, \ldots$ horizon $T$
- Value function $V(t, x) = \sup_{\tau \leq T} \mathbb{E}_{t,x}[h(\tau, X_{\tau})]$
- Optimization is over stopping times $\tau$
- **Solution:** $\tau^* = \inf\{t : X_t \in \mathcal{S}_t\} \land T$. Stopping region:
  $\mathcal{S}_t = \{x : V(t, x) = h(t, x)\}$
- eg $(X_t)$ is GBM; $h(t, x) = e^{-rt}(K - x)_+ = \text{Bermudan Put}$
Stopping Rule via Timing Value

\[ T(t, x) := \mathbb{E}_{t,x}[V(t + 1, X_{t+1})] - h(t, x) = \mathbb{E}_{t,x}[h(\tau_{t+1}, X_{\tau_{t+1}})] - h(x). \]

- Stopping decision is characterized by \( \mathcal{S}_t = \{ x : T(t, x) < 0 \} \)
- To find \( \tau^* \), it’s sufficient to evaluate the conditional expectation, i.e. **approximate the sign** of \( T(t, \cdot) \) for \( t = T - 1, T - 2, \ldots, 0 \)
Conditional Expectation

\[ f(x) := \mathbb{E}[h(X.)|X_0 = x]. \]

- Input: Markov process \( X \) with state space \( \mathcal{X} \) & (path-) Functional \( h(X.) \)
- Output: the conditional mean map \( x \mapsto f(x) \)
- Generalizes the problem of \emph{pointwise} estimates at a fixed \( x \)
- Appears as a building block:
  - Optimal switching/impulse control
  - XVA
  - BSDEs
  - Capital Requirements/Insurance
Regression Monte Carlo

\[ f(x) := \mathbb{E}[g(X_\cdot)|X_0 = x]. \]

- Classical Monte Carlo for a fixed \( x_0 \): \( \hat{f}(x_0) := \frac{1}{N} \sum_{n=1}^{N} h(x^n) \) where \( x^n \) are \( N \) simulated paths
- Need to be able to predict \( f(x) \) for any \( x \in \mathcal{X} \)
- The state space \( \mathcal{X} \) is multi-dimensional and continuous
  \( \rightarrow \) Construct a grid \( x_1^{1:N} \) and borrow information spatially
- Statistical regression: smooth + interpolate
RMC for Optimal Stopping

- Backwards induction in time ($\mathcal{S}_T = \{x : h(t, x) \geq 0\}$)
- **Given** stopping sets: $\hat{\mathcal{S}}_{t+1:T}$
- Starting at $X_t = x$, simulate trajectory $X_{t+1:T}^x$ and take $\tau' = \inf\{s > t : X_s^x \in \hat{\mathcal{S}}_s\}$
- Pathwise future payoff $y := h(\tau', X_{\tau'}^x)$ satisfies

$$
\mathbb{E}_{t,x}[Y_x] = C(t, x) \iff Y_x = C(t, x) + \varepsilon(x)
$$

where $C(t, x) = T(t, x) + h(t, x)$ is the **continuation value**

- Now generate a stochastic grid $(x^n_t)_{n=1}^N$ and paths $x^{1:N}_{t+1:T}$
- Obtain a sample $\{x_t, y_t\}_{1}^{N}$
- **Estimate** $\hat{C}(t, \cdot)$ and set $\hat{\mathcal{S}}_t := \{x : \hat{C}(t, x) - h(t, x) < 0\}$
- Popularized by Longstaff & Schwartz (2001)
Metamodeling

**AIM**: Build an approximation of \( \hat{C}(t, \cdot) \)
- Choose an approximation architecture \( \mathcal{H} \) and loss function \( L \)
- Generate the grid \( x_t^{1:N} \): Experimental Design
- Set \( \hat{C}(t, \cdot) = \arg \min_{C \in \mathcal{H}} L(C; (x, y)^{1:N}) \)
- Repeat over \( t = T - 1, T - 2, \ldots \)

**Traditionally:**
- Data is generated using the transition density of \( X \) (“path-simulation”)
- Least-Squares parametric regression, i.e.
  \( \mathcal{H} = \text{span}(B_i(x), i = 1, \ldots, r) \)
- (The implied loss function is \( \mathbb{E}_{0,X_0}[\{(\hat{C}(X_t) - C(X_t))^2\}] \))
What is Metamodelling?

- Classical regression – data is given and try to fit the “best curve”
- In metamodelling generating data (through efficient simulations) is part of the solution
- Also, typically look for a non-parametric model (dense $\mathcal{H}$)
- Goes by many other names: response surface modeling, statistical learning, DACE (design and analysis of computer experiments), emulation
- Used extensively in machine learning; simulation optimization, computational statistics
- Connects to CS, OR, stats communities (language barriers!)
Improving RMC

- Main concerns are **Speed/memory** – convergence of RMC is slow; often need $\gg 10^5$ paths to obtain a good estimate
- Desire ability to handle a “**black-box**” setting, e.g. 5-D system with implicit dynamics, and limited known structure
- Timing optionality is now embedded in a ton of contracts – wish to have a “universal” algorithm
- Traditional methods offer few performance guarantees (eg. sensitive to the choice of basis functions) and are hard to trust
Contributions

- There has been extensive ongoing research on better regressions: Belomestny, Bouchard, Gobet, Kohler, Oosterlee, Stentoft, Tompaidis, ...

Contributions

- There has been extensive ongoing research on better regressions: Belomestny, Bouchard, Gobet, Kohler, Oosterlee, Stentoft, Tompaidis, ...
- **Contribution 1**: investigate impact of RMC experimental designs and suggest several (improved) choices
- **Contribution 2**: propose use of kriging metamodels
- RMC is often called Least Squares Monte Carlo. This puts misplaced narrow emphasis on a specific regression framework, and tends to ignore the design aspect. We advocate a shift in terminology to better align with the underlying problem.
Modeling Conditional Expectation

\[ f(x) := \mathbb{E}[g(X) | X_0 = x]. \]

Must impose some structure on \( f \) (\( X \) is a "nice" process, so \( f \) is "smooth")

- Project onto basis functions: \( f(x) = \sum_{i=1}^{R} a_i H_i(x) \)
- Smoothing spline (piecewise cubic)
- Piecewise linear
- Piecewise constant \( f(x) = \sum_i a_i \mathbb{1}_{\{x \in R_i\}} \)
- Fully nonparametric (kernel): \( f(x) = \sum_i K(x, x^i)y^i \)
- Gaussian process
Stochastic Kriging

- Data-generating process $Y(x) = C(t, x) + \varepsilon(x)$ where $\varepsilon(x) \sim N(0, \sigma^2(x))$
- Assume the continuation value $C(t, \cdot)$ lives in the function space $\mathcal{H}_K$ – Gaussian RKHS
- Means $C(t, \cdot)$ is a realization of a Gaussian random field with a covariance structure defined by $K$, $\mathcal{H} = \text{span}(K(\cdot, x) : x \in \mathcal{X})$
- $K(x, x') := \mathbb{E}[f(x)f(x')]$ controls the spatial decay of correlation, i.e. smoothness of $C(t, \cdot)$
- e.g. Gaussian kernel $K(x, x') = \tau^2 \exp(-\|x - x'\|^2/\theta^2)$ – elements of $\mathcal{H}_K$ are $C^\infty$, with lengthscale $\theta$ and fluctuation scale $\tau$.
- Use $L^2$ projection: $\hat{C}(t, \cdot) = \arg \min_{C \in \mathcal{H}} \sum_{i=1}^{N} (C(x^i) - y^i)^2$
- Representer theorem implies that $\hat{C}(t, x) = \sum_{i=1}^{N} w_i K(x, x^i)$
Stochastic Kriging

- Think of $C(t, \cdot)$ as a random element in $\mathcal{H}_K$ with a Gaussian prior $C(t, x) \sim N(0, \tau^2)$
- The posterior conditional on $G \equiv (x, y)^{1:N}$ is also Gaussian
- Marginally $C(t, x)|G \sim N(m(x), v^2(x))$

$$m(x) = \tilde{k}(x)^T(K + \Sigma)^{-1}\tilde{y}$$
$$v(x, x') = K(x, x') - \tilde{k}(x)^T(K + \Sigma)^{-1}\tilde{k}(x')$$

$$K_{ij} = K(x^i, x^j), \Sigma = diag(\sigma^2(x^i)), k_i = K(x, x^i)$$
- Linear model in the infinite basis expansion defined by $K$
Kriging Example 1

- The posterior is a measure on $\mathcal{H}_K$ (i.e. function-valued).
- Visually has a “football” shape—$v^2(x)$ has local minima at $x^i$'s.
- The mean $m(x)$ is a linear combination of kernel eigenfunctions centered at design sites.
- Outside the domain $\mathcal{X}'$, revert to prior $m(x) \rightarrow 0$, $v^2(x) \rightarrow \tau^2$.
- Below: $\theta = 2$, $\tau = 1.5$, $\sigma^2(x) \equiv 0.2^2$.
Kriging Example 2

- **Global consistency** – converge to the truth as $N \to \infty$
- **Optimized Matern-5/2 kernel**

$$K(x, x'; \tau, \theta) = \tau^2 (1 + (\sqrt{5} + 5/3)\|x - x'\|_\theta^2) \cdot e^{-\sqrt{5}\|x - x'\|_\theta}$$
Fitting a GP

- Need to know the kernel hyperparameters – \( \tau, \theta \), et cetera. Use MLE (nonlinear optimization problem).
- \( \theta \) is the lengthscale – correlation decay
- \( \tau^2 \) is the process variance – has analytic MLE once \( \theta \) is known
- GP is expensive compared to e.g LM; complexity is \( O(N^3) \) for a design of size \( N \)
- Allows a lot of analytic formulas to understand the fit and its uncertainty
- Kriging is becoming the gold standard in the simulation/DACE communities
- Used DiceKriging package in R – off-the-shelf use
Simulation Noise

Figure: Left: scatterplot of \((x, H_t(X^x) - h(t, x))\) over 10,000 distinct \(x \in \mathbb{R}_+\). Right: Histogram of \(N = 200\) pathwise future payoffs \(y^n \sim H_t(X^x)\) starting at \(x = 35\) in a 1-D Bermudan Put problem; \(t = 0.6\). The vertical dashed line indicates the empirical mean \(\mathbb{E}[H_t(X^x) | X_t = 35] \simeq \text{Ave}(y^{1:N}) = 5.49\). Note that in 24 out of 200 scenarios, the payoff \(y^n\) was zero, creating a point mass in the distribution of \(H_t(X^x)\) and generating a significant negative skew. Other moments were \(\text{StDev}(y^{1:N}) = 2.45, \text{Skew}(y^{1:N}) = -1.28\) and \(\text{Max}(y^{1:N}) = 9.87\).
Simulation Noise

- Knowing the distribution of simulation noise $\varepsilon(x)$ is fundamental for meta-modeling.
- Simulation noise is highly state-dependent in RMC.
- Also, distribution can be skewed/far from Gaussian.
- **Solution 1**: treat it as a constant $\sigma^2$ (so-called “nugget”), can estimate along with other kernel hyper-parameters.
- **Solution 2**: build an empirical estimate through replicating simulations at a fixed site $x$.
  (Resembles a Monte Carlo forest)
- **Solution 3**: model $x \mapsto \sigma^2(x)$ via an auxiliary metamodel.
Batching

- Generate $M$ independent realizations $y^{(i)} \sim Y_x$ of pathwise payoffs starting at $X_t = x$
- Set the average $\bar{y}(x) = \frac{1}{M} \sum_{i=1}^{M} y^{(i)}(x)$
- Empirical $\tilde{\sigma}^2(x) := \frac{1}{M-1} \sum_{i=1}^{M} (y^{(i)}(x) - \bar{y}(x))^2$
- The averaged simulations still follow the same statistical model but with signal-to-noise ratio improved by factor of $M$
- Size of macro-design $\mathcal{Z}'$ is $N/M$ — much faster fitting
- Also, $\bar{Y}$ has almost-Gaussian simulation noise
Batched Kriging Metamodel for $T(t, \cdot)$

**Figure**: LHS design $\mathcal{Z}$ of size $N = 3000$ with $M = 100$ replications. The vertical “error” bars indicate the 95% quantiles of the simulation batch at $x$, while the dotted lines indicate the 95% credibility interval (CI) of the kriging metamodel fit.
Deterministic Kriging

- If $M$ is very large, $\bar{\sigma}^2(x)/M \sim 0$ and can view $\bar{Y}_x$ as deterministic.
- Metamodel becomes an interpolator.

**Figure:** The boxplots summarize the distribution of $y^{(m)}(x^n)$'s, $m = 1, \ldots, M = 1600$. The dots indicate the batch means $\bar{y}(x^n)$ which are exactly interpolated by the two meta-models. $\mathcal{Z}' = \{30, 32, 34, 35, 36, 38\}$. 

$Z' = \{30, 32, 34, 35, 36, 38\}$. 

$Ludkovski Adaptive RMC$
Regression Designs

Based on $S_t \mid S_0$

Uniform in $[30, 40]$

Monte Carlo forest

Adaptive Grid
Experimental Design

- The meta-model should learn $C(t, \cdot) - Z$ should cover the domain $\mathcal{X}$
- Space-filling designs – lattice, low-discrepancy (Sobol)
- **LHS** Latin Hypercube sampling: random space-filling
- User must specify the effective $\mathcal{X}'$ (typically a rectangle)
Experimental Design

- The meta-model should learn $C(t, \cdot) – Z$ should cover the domain $\mathcal{X}$
- Space-filling designs – lattice, low-discrepancy (Sobol)
- **LHS** Latin Hypercube sampling: random space-filling
- User must specify the effective $\mathcal{X}'$ (typically a rectangle)
- The design should reflect the underlying $(X_t)$
- **Empirical** sampling: $Z$ is constructed by drawing from $X_t$
- Automatically has the right “shape”
- This is the standard approach. Sensitive to $X_0$ (e.g. OTM Puts)

(Optimal Design is **NP-Hard** so heuristics are common)
LHS \( M = 20, N' = 150 \)  LHS \( M = 100, N' = 30 \)  Emp \( M = 100, N' = 30 \)

**Figure:** Three different designs for fitting a kriging metamodel of the continuation value. *Top* panels show the fitted \( \hat{T}(t, \cdot) \) as well as the distinct design sites \( x^{1:N'} \). *Middle* panels plot the corresponding surrogate standard deviation \( \nu(x) \). *Bottom* panels display the loss metric \( \ell(x; \mathcal{Z}) \).
Adaptive Design

- Recall that aim to learn the sign of $T(t, \cdot)$
- Gradually grow $Z^{(k)}$, $k = N_0, \ldots, N$
- Add new locations greedily according to acquisition function
  
  \[ x^{k+1} = \arg \max \text{EI}_k(x) \]

- Favor points where $m^{(k)}(x) \approx 0$ (close to zero-contour) or $\nu^{(k)}(x)$ is large (reduce uncertainty)

- Loss from making the wrong stopping decision at $(t, x)$ is

  \[
  \ell(x; Z) := \int_{\mathbb{R}} |y - h(t, x)| 1\{m(x) < h(t, x) < y \cup y < h(t, x) < m(x)} \mathcal{M}_x(dy)
  \]

- Analytic expression for
  \[\text{EI}_k(x) := \mathbb{E} [\ell^{(k)}(x) - \ell^{(k+1)}(x) | Z^{(k)}, x^{k+1} = x] \]

- ZC-SUR strategy: maximizes stepwise expected reduction in loss

- See Gramacy-L. (SIFIN 2015)
Sequential Design: $K = 20$

Initialize with a LHS design $\mathcal{Z}^{(20)}$
Sequential Design: $K = 30$

Zoom to the stopping boundary
Sequential Design: $K = 40$

Prefer regions that are more likely for $X_t$
Optimal Stopping for a 2D Stoch Vol Model

Figure: Adaptive vs LHS designs. Bermudan Put $e^{-rt}(100 - X_1)_+$ with a Heston SV model. Both designs used $N = 10000$ simulations. Color-coded according to $T(t, x)$; contour indicates the stopping boundary.
## Comparison in 1-D GBM Put

<table>
<thead>
<tr>
<th>Batch Size</th>
<th>LHS Spline</th>
<th>LHS Kriging</th>
<th>Emp Kriging</th>
<th>Seq Kriging</th>
</tr>
</thead>
<tbody>
<tr>
<td>$M = 3$</td>
<td>2.306</td>
<td>2.304</td>
<td>2.306</td>
<td>2.303</td>
</tr>
<tr>
<td>$M = 8$</td>
<td>2.306</td>
<td>2.306</td>
<td>2.308</td>
<td>2.305</td>
</tr>
<tr>
<td>$M = 20$</td>
<td>2.292</td>
<td>2.305</td>
<td>2.286</td>
<td>2.295</td>
</tr>
<tr>
<td>$M = 50$</td>
<td>2.302</td>
<td>2.303</td>
<td>2.302</td>
<td>2.309</td>
</tr>
<tr>
<td>$M = 100$</td>
<td>2.302</td>
<td>2.303</td>
<td>2.304</td>
<td><strong>2.311</strong></td>
</tr>
<tr>
<td>$M = 250$</td>
<td>2.304</td>
<td>2.304</td>
<td>2.303</td>
<td>2.309</td>
</tr>
</tbody>
</table>

**Table:** Performance of different DoE approaches to RMC in the 1-D Bermudan Put setting, $h(t, x) = e^{-rt}(40 - x)_+$. All methods utilize $|Z_t| = 3000$. The LHS input space was $\tilde{X} = [25, 40]$. Results are based on averaging 100 runs of each method, and evaluating $V(0, X_0)$ on a fixed out-of-sample database of $N_{out} = 50,000$ scenarios.
## 2D Examples

<table>
<thead>
<tr>
<th>Method</th>
<th>$\hat{V}(0, X_0)$ (StDev.)</th>
<th>#Sims</th>
<th>Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>Brockwell Rhambarat SV5</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>LSM $N = 5 \cdot 10^4$</td>
<td>15.98 (0.04)</td>
<td>2.5 $\cdot 10^6$</td>
<td>24</td>
</tr>
<tr>
<td>LSM $N = 1.25 \cdot 10^5$</td>
<td>16.38 (0.03)</td>
<td>6.25 $\cdot 10^6$</td>
<td>52</td>
</tr>
<tr>
<td>LHS km $N = 2500$</td>
<td>16.07 (0.16)</td>
<td>1.07 $\cdot 10^6$</td>
<td>25</td>
</tr>
<tr>
<td>LHS km $N = 10000$</td>
<td>16.48 (0.06)</td>
<td>4.8 $\cdot 10^6$</td>
<td>168</td>
</tr>
<tr>
<td>SUR km $N = 4000$</td>
<td>16.42 (0.11)</td>
<td>1.67 $\cdot 10^6$</td>
<td>65</td>
</tr>
<tr>
<td>Agrawal, Juneja and Sircar</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>LSM $N = 5 \cdot 10^4$</td>
<td>18.63 (0.03)</td>
<td>1.0 $\cdot 10^6$</td>
<td>25</td>
</tr>
<tr>
<td>LSM $N = 1.25 \cdot 10^5$</td>
<td>18.81 (0.02)</td>
<td>2.5 $\cdot 10^6$</td>
<td>60</td>
</tr>
<tr>
<td>LHS km $N = 2500$</td>
<td>18.79 (0.04)</td>
<td>0.20 $\cdot 10^6$</td>
<td>11</td>
</tr>
<tr>
<td>LHS km $N = 10000$</td>
<td>18.88 (0.02)</td>
<td>0.81 $\cdot 10^6$</td>
<td>53</td>
</tr>
<tr>
<td>SUR km $N = 4000$</td>
<td>18.86 (0.02)</td>
<td>0.35 $\cdot 10^6$</td>
<td>64</td>
</tr>
<tr>
<td>SUR km $N = 10000$</td>
<td><strong>18.90</strong> (0.01)</td>
<td>0.80 $\cdot 10^6$</td>
<td>103</td>
</tr>
</tbody>
</table>
Kriging Performance

- Kriging appears very promising as a flexible, off-the-shelf regression framework.
- Gives smooth, non-parametric fits for $C(t, \cdot)$.
- Emphasizes the interpolation vs. smoothing aspect of metamodeling.
- Easy implementation via public R packages.
- Order of magnitude slower than a Least-Squares model (not important if simulations are the bottleneck).
Experimental Design Performance

- **Batching** has minimal effect on performance (but major effect on speed)
- (Random) space-filling designs allow to reduce size of design by a factor of 3-5
- Compared to standard LSM this reduces simulation budget by 25-50%
- Adaptive sequential designs
  - Yield **further** substantial savings (up to an order of magnitude)
  - Significant regression overhead as must fit multiple (kriging) metamodels
  - Worth it if in high dimensions $d > 3$ and simulation budget is very constrained
Bermudan Max Call \((\max(X_1, X_2) - K)_+\)

- It is often nontrivial to specify a good domain \(X'\)
- This is the advantage of the empirical design
- Sequential designs really begin to shine
The Future

- Finding conditional expectations is a **metamodeling** problem
- Can squeeze (a lot) of **extra efficiency** by jointly targeting experimental design + regression
- Lots more opportunities in this direction
- e.g. adapt to the BSDE numerical algorithms (Bender, Gobet)
- Also more general control problems (**optimal switching**, sequential games, et cetera)

\[
\text{RMC} = + 
\]
The Future

- Finding conditional expectations is a *metamodelling* problem
- Can squeeze (a lot) of *extra efficiency* by jointly targeting experimental design + regression
- Lots more opportunities in this direction
- e.g. adapt to the BSDE numerical algorithms (Bender, Gobet)
- Also more general control problems (*optimal switching*, sequential games, et cetera)

\[
\text{RMC} = \text{Regression} + \text{Stochastic Grid}
\]
The Future

- Finding conditional expectations is a **metamodelling** problem
- Can squeeze (a lot) of **extra efficiency** by jointly targeting experimental design + regression
- Lots more opportunities in this direction
- e.g. adapt to the BSDE numerical algorithms (Bender, Gobet)
- Also more general control problems (**optimal switching**, sequential games, et cetera)

RMC = Regression + Stochastic Grid

THANK YOU!
References


R. Gramacy and M. Ludkovski *Sequential Design for Optimal Stopping Problems* SIFIN 6(1), 2015, pp. 748–775


arXiv:1509.00980
Require: $N$ – number of initial grid points
1: $\mathcal{S}_T \leftarrow \mathcal{X}$
2: for $t = T - 1, T - 2, \ldots, 0$ do
3: \hspace{1em} $k \leftarrow 0$
4: \hspace{1em} Generate an initial grid $\{x_t^{1:N}\}$, and corresponding classifier $\mathcal{S}_t^{(0)}$
5: \hspace{1em} while the current grid needs refining do
6: \hspace{2em} $k \leftarrow k + 1$
7: \hspace{2em} Generate new grid point(s) $\{x_t^{(k),n'}\}$ $n' = 1, \ldots, N^{(k)}$
8: \hspace{2em} Simulate forward trajectories $X_t^{(k),1:N^{(k)}}$. Using $\hat{\mathcal{S}}_{t+1:T}$ find $y_t^{(k),1:N^{(k)}}$
9: \hspace{2em} Update the classifier to $\mathcal{S}_t^{(k)}$ using new samples $(x_t^{(k)}, y_t^{(k)})^{1:N^{(k)}}$
10: \hspace{2em} (Update the classifiers $\hat{\mathcal{S}}_{t+1:T-1}$ using $x_{t+1:T-1}^{(k),1:N^{(k)}}$)
11: \hspace{2em} Save the overall grid $\{x_t\} \leftarrow \{x_t\} \cup \{x_t^{(k),1:N^{(k)}}\}$
12: \hspace{1em} end while
13: \hspace{1em} Generate final estimate of the classifier at time step $t$, $\hat{\mathcal{S}}_t$
14: end for
15: Simulate forward trajectories $X^n_0:T$ from $X^n = x_0$ using $\hat{\mathcal{S}}_{0:T}$
16: return $V(0, x_0) \approx \frac{1}{N} \sum_{n=1}^{N} h_\tau^n (X^n_\tau^n)$
17: return Estimated policy $\{\hat{\mathcal{S}}_{0:T}\}$. 
Sequential Design for Regression Monte Carlo

Generate the grids adaptively online. [Vanilla RMC re-uses the grids during forward simulations. We regenerate fresh paths at each step]

- Start with initial grid $Z^{(n_0)} \equiv \{x^1_{t:n_0}\}$
- Build initial approximation $G^{(n_0)}_t$
- LOOP for $k = n_0, n_0 + 1, \ldots$
  - Identify promising regions
  - Generate new data $\{x^{k+1}_{t:T}\}$ and costs-to-go
    $y^{k+1}_t = h(x^k_{t:k+1}) - h(x^{k+1}_t)$.
  - Update the fit to $G^{(k+1)}_t$
- END LOOP
- Repeat above at each time-step $t = T - 1, \ldots, 1$