Déjà Vu All Over Again

Efficiency when Financial Simulations are Repeated

Jeremy Staum

Dept. of Industrial Engineering and Management Sciences
McCormick School of Engineering
Northwestern University

22 March 2010
Fields Workshop on Computational Methods in Finance

Objectives and Outline

Objectives

1. vision for research in financial simulation
2. overview of resources for achieving it

Objectives and Outline

Objectives

1. vision for research in financial simulation
2. overview of resources for achieving it

Outline

1. repeated simulations: examples and paradigms
2. methodological resources
3. examples of applied research
4. future directions

Paradigms for Repeated Financial Simulations

Run-oriented paradigm: one run per expectation approximated

On each day $i = 1, 2, \ldots$, for each security $j = 1, \ldots, J$,
run a simulation to approx. $\mu(\theta_i, \psi_j)$ by $\frac{1}{n}\sum_{h=1}^{n} Y(\omega_{ijh}, \theta_i, \psi_j)$.

SIMULATION EFFORT = COMPUTATIONAL <u>EXPENSE</u>

Paradigms for Repeated Financial Simulations

Run-oriented paradigm: one run per expectation approximated

On each day $i = 1, 2, \ldots$, for each security $j = 1, \ldots, J$,
run a simulation to approx. $\mu(\theta_i, \psi_j)$ by $\frac{1}{n}\sum_{h=1}^{n} Y(\omega_{ijh}, \theta_i, \psi_j)$.

SIMULATION EFFORT = COMPUTATIONAL <u>EXPENSE</u>

Day 1: Security 1, Security 2: déjà vu

Paradigms for Repeated Financial Simulations

Run-oriented paradigm: one run per expectation approximated

On each day $i = 1, 2, \ldots$, for each security $j = 1, \ldots, J$,
run a simulation to approx. $\mu(\theta_i, \psi_j)$ by $\frac{1}{n}\sum_{h=1}^{n} Y(\omega_{ijh}, \theta_i, \psi_j)$.

SIMULATION EFFORT = COMPUTATIONAL <u>EXPENSE</u>

Day 1: Security 1, Security 2: déjà vu
Day 2: déjà vu all over again

data is wasted; no learning

Paradigms for Repeated Financial Simulations

Run-oriented paradigm: one run per expectation approximated

On each day $i = 1, 2, \ldots$, for each security $j = 1, \ldots, J$,
run a simulation to approx. $\mu(\theta_i, \psi_j)$ by $\frac{1}{n}\sum_{h=1}^{n} Y(\omega_{ijh}, \theta_i, \psi_j)$.

SIMULATION EFFORT = COMPUTATIONAL <u>EXPENSE</u>

Day 1: Security 1, Security 2: déjà vu
Day 2: déjà vu all over again

data is wasted; no learning

Problem-oriented paradigm: one experiment per problem

1. Perform an experiment with multiple simulation runs.
2. Use results to approx. any $\mu(\theta, \psi)$.

SIMULATION EFFORT = COMPUTATIONAL <u>INVESTMENT</u>

Goals: reduce computational cost, simulation on demand

Fixed Sets of Tasks

Example (many options)

Value k options differing only in $\psi =$ (strike, maturity), want to know $\mu(\psi_1), \ldots, \mu(\psi_k)$.

Run-oriented paradigm:

For all j, simulate paths $\omega_{j1}, \ldots, \omega_{jn}$, and approximate $\mu(\psi_j)$ by $\frac{1}{n}\sum_{h=1}^{n} Y(\omega_{jh}, \psi_j)$.

Fixed Sets of Tasks

Example (many options)

Value k options differing only in $\psi =$ (strike, maturity), want to know $\mu(\psi_1), \ldots, \mu(\psi_k)$.

Run-oriented paradigm:

For all j, simulate paths $\omega_{j1}, \ldots, \omega_{jn}$, and approximate $\mu(\psi_j)$ by $\frac{1}{n} \sum_{h=1}^{n} Y(\omega_{jh}, \psi_j)$.

A standard problem-oriented efficiency technique: reuse paths.

1. Simulate paths $\omega_1, \ldots, \omega_n$.
2. For all $j = 1, \ldots, k$, approx. $\mu(\psi_j)$ by $\frac{1}{n} \sum_{h=1}^{n} Y(\omega_h, \psi_j)$.

Fixed Sets of Tasks

Example (many options)

Value k options differing only in $\psi =$ (strike, maturity), want to know $\mu(\psi_1), \ldots, \mu(\psi_k)$.

Run-oriented paradigm:

For all j, simulate paths $\omega_{j1}, \ldots, \omega_{jn}$, and approximate $\mu(\psi_j)$ by $\frac{1}{n}\sum_{h=1}^{n} Y(\omega_{jh}, \psi_j)$.

A standard problem-oriented efficiency technique: reuse paths.

1. Simulate paths $\omega_1, \ldots, \omega_n$.
2. For all $j = 1, \ldots, k$, approx. $\mu(\psi_j)$ by $\frac{1}{n}\sum_{h=1}^{n} Y(\omega_h, \psi_j)$.

Even more problem-oriented:

To approx. $\mu(\psi_j)$, also use simulation runs with $\psi \neq \psi_j$.

Sequence of Repeated Tasks

Example (moving markets)

Value a security every day, given newly calibrated model parameters Θ: approx. $\mu(\Theta_1), \mu(\Theta_2), \ldots$

Run-oriented paradigm:

Each day i, simulate paths given Θ_i, use them to approx. $\mu(\Theta_i)$.

Sequence of Repeated Tasks

Example (moving markets)

Value a security every day, given newly calibrated model parameters Θ: approx. $\mu(\Theta_1), \mu(\Theta_2), \ldots$

Run-oriented paradigm:

Each day i, simulate paths given Θ_i, use them to approx. $\mu(\Theta_i)$.

Problem-oriented paradigm:

1. Perform simulations conditional on $\theta_1, \ldots, \theta_k$; store some information.
2. Use it in approximating $\mu(\Theta_i)$.

Nested Simulation I

Example (portfolio risk measurement)

1. Sample scenarios $\Theta_1, \ldots, \Theta_K$.
2. In each scenario, approx. portfolio value $\mu(\Theta_i)$ by $\hat{\mu}(\Theta_i)$.
3. Evaluate the risk measure on $\hat{\mu}(\Theta_1), \ldots, \hat{\mu}(\Theta_k)$.

Run-oriented paradigm (step 2):

For each i, run simulation conditional on Θ_i to get $\hat{\mu}(\Theta_i)$.

Nested Simulation I

Example (portfolio risk measurement)

1. Sample scenarios $\Theta_1, \ldots, \Theta_K$.
2. In each scenario, approx. portfolio value $\mu(\Theta_i)$ by $\hat{\mu}(\Theta_i)$.
3. Evaluate the risk measure on $\hat{\mu}(\Theta_1), \ldots, \hat{\mu}(\Theta_k)$.

Run-oriented paradigm (step 2):

For each i, run simulation conditional on Θ_i to get $\hat{\mu}(\Theta_i)$.

Problem-oriented paradigm (step 2):

- Run simulations conditional on each $\theta_1, \ldots, \theta_k$ where $k \ll K$.
- Use them in approximating $\mu(\Theta_i)$ for $i = 1, \ldots, K$.

bias

Frye (1998), "Monte Carlo by Day."

Nested Simulation II

Example (American option pricing)

1. Simulate paths $S_1^{(i)}, \ldots, S_T^{(i)}$ for $i = 1, \ldots, n$.
2. Approx. continuation value $C(t, S_t^{(i)})$ for each step and path.
3. When to exercise on each path? $\hat{\tau}_i$.
4. Approx. price by $\frac{1}{n}\sum_{i=1}^{n} Y(\hat{\tau}, S_{\hat{\tau}_i}^{(i)})$.

Run-oriented paradigm (step 2):

For each step and path, conditional on steps $1, \ldots, t$ of path i, run simulations to approximate $C(t, S_t^{(i)})$, discard them.

Nested Simulation II

Example (American option pricing)

1. Simulate paths $S_1^{(i)}, \ldots, S_T^{(i)}$ for $i = 1, \ldots, n$.
2. Approx. continuation value $C(t, S_t^{(i)})$ for each step and path.
3. When to exercise on each path? $\hat{\tau}_i$.
4. Approx. price by $\frac{1}{n} \sum_{i=1}^{n} Y(\hat{\tau}, S_{\hat{\tau}_i}^{(i)})$.

Run-oriented paradigm (step 2):

For each step and path, conditional on steps $1, \ldots, t$ of path i, run simulations to approximate $C(t, S_t^{(i)})$, discard them.

Problem-oriented paradigm (step 2):

Use all time steps of all paths to approximate each $C(t, S_t^{(i)})$.

Regression Monte Carlo: problem-oriented

(Longstaff&Schwartz; Tsitsiklis&Van Roy)

Choose basis functions b (vector-valued).

Backward recursion to approx. the continuation values $C(t, S_t^{(i)})$:

For $i = 1, \ldots, n$, $\hat{C}(T, S_T^{(i)}) = 0$.

For $t = T, \ldots, 1$,

1. $\hat{V}(t, S_t^{(i)}) = \max\{Y(t, S_t^{(i)}), \hat{C}(t, S_t^{(i)})\}$.
2. Multiple regression of $\hat{V}(t, S_t^{(1)}), \ldots, \hat{V}(t, S_t^{(n)})$ on $b(S_{t-1}^{(1)}), \ldots, b(S_{t-1}^{(n)})$ yields $\hat{\beta}_t$.
3. $\hat{C}(t-1, S_{t-1}^{(i)}) = b(S_{t-1}^{(i)})\hat{\beta}_t$.

Regression Monte Carlo: problem-oriented

(Longstaff&Schwartz; Tsitsiklis&Van Roy)

Choose basis functions b (vector-valued).

Backward recursion to approx. the continuation values $C(t, S_t^{(i)})$:

For $i = 1, \ldots, n$, $\hat{C}(T, S_T^{(i)}) = 0$.

For $t = T, \ldots, 1$,

1. $\hat{V}(t, S_t^{(i)}) = \max\{Y(t, S_t^{(i)}), \hat{C}(t, S_t^{(i)})\}$.
2. Multiple regression of $\hat{V}(t, S_t^{(1)}), \ldots, \hat{V}(t, S_t^{(n)})$ on $b(S_{t-1}^{(1)}), \ldots, b(S_{t-1}^{(n)})$ yields $\hat{\beta}_t$.
3. $\hat{C}(t-1, S_{t-1}^{(i)}) = b(S_{t-1}^{(i)})\hat{\beta}_t$.

Bias depends on goodness of fit,
which depends on choice of basis functions (problem-specific).

Regression Monte Carlo: problem-oriented

(Longstaff&Schwartz; Tsitsiklis&Van Roy)

Choose basis functions b (vector-valued).

Backward recursion to approx. the continuation values $C(t, S_t^{(i)})$:

For $i = 1, \ldots, n$, $\hat{C}(T, S_T^{(i)}) = 0$.

For $t = T, \ldots, 1$,

1. $\hat{V}(t, S_t^{(i)}) = \max\{Y(t, S_t^{(i)}), \hat{C}(t, S_t^{(i)})\}$.
2. Multiple regression of $\hat{V}(t, S_t^{(1)}), \ldots, \hat{V}(t, S_t^{(n)})$ on $b(S_{t-1}^{(1)}), \ldots, b(S_{t-1}^{(n)})$ yields $\hat{\beta}_t$.
3. $\hat{C}(t-1, S_{t-1}^{(i)}) = b(S_{t-1}^{(i)})\hat{\beta}_t$.

Bias depends on goodness of fit,
which depends on choice of basis functions (problem-specific).

Essential idea: <u>metamodel</u> $\hat{C}(t, \cdot)$ of $C(t, \cdot)$.

Metamodeling

Run-oriented: to learn about $\mu(\theta)$, run the simulation model at θ.

Problem-oriented metamodeling:

To learn about the function μ,

1. Perform a simulation experiment with runs at $\theta_1, \ldots, \theta_k$.
2. Use simulation output to approx. μ by the metamodel $\hat{\mu}$.

Metamodeling

Run-oriented: to learn about $\mu(\theta)$, run the simulation model at θ.

Problem-oriented metamodeling:

To learn about the function μ,

1. Perform a simulation experiment with runs at $\theta_1, \ldots, \theta_k$.
2. Use simulation output to approx. μ by the metamodel $\hat{\mu}$.

The metamodel's $\hat{\mu}(\theta)$ is faster but less accurate
than a long simulation run at θ.

Goals of metamodeling:

- reduce computational cost
- simulation on demand (Monte Carlo by day)
- Greeks from $\nabla \hat{\mu}$

http://users.iems.northwestern.edu/~staum/MonteCarloFinance.pdf

Metamodeling: there ain't no such thing as a free lunch

Inference about $\mu(\theta)$ without simulating at θ needs assumptions:

- about spatial variability in μ
- about noise in simulation output

Simulation output at θ_i with n replications is

$$Y(\theta; n) = \mu(\theta) + \frac{1}{n}\sum_{j=1}^{n} \varepsilon_j(\theta).$$

Beware metamodel misspecification which causes bad $\hat{\mu}$.

http://www.informs-sim.org/wsc09papers/011.pdf

http://users.iems.northwestern.edu/~nelsonb/SK/StaumTutorialWSC09.pdf

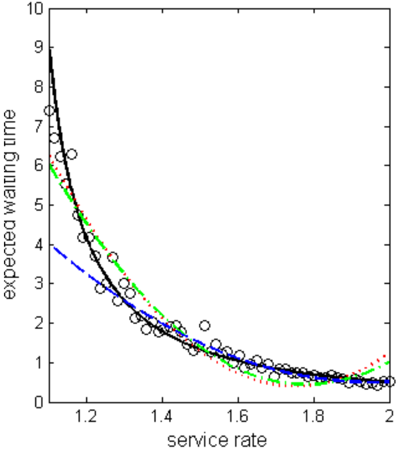
Regression Metamodeling?

Assumptions:

- $\mu(\theta) = b(\theta)\beta$ for known b and some β.
- All $\varepsilon_j(\theta)$ are independent.
- OLS: $\varepsilon_j(\theta) \sim N(0, v)$.
- WLS: $\varepsilon_j(\theta) \sim N(0, v(\theta))$.

Legend: (quadratic metamodels)

- black line = truth, $\circ$ = data
- fit to data: OLS, WLS
- red dots = best fit to truth



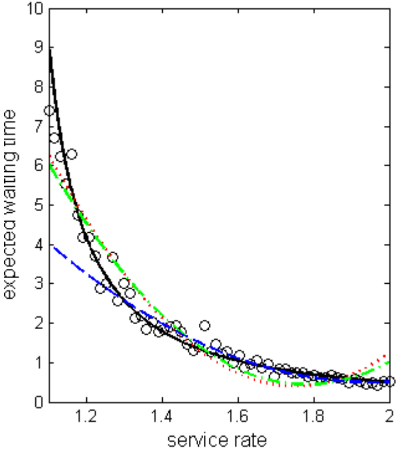
Regression Metamodeling?

Assumptions:

- $\mu(\theta) = b(\theta)\beta$ for known b and some β.
- All $\varepsilon_j(\theta)$ are independent.
- OLS: $\varepsilon_j(\theta) \sim N(0, v)$.
- WLS: $\varepsilon_j(\theta) \sim N(0, v(\theta))$.

Legend: (quadratic metamodels)

- black line = truth, $\circ$ = data
- fit to data: OLS, WLS
- red dots = best fit to truth



Misspecification: best fit may be bad.

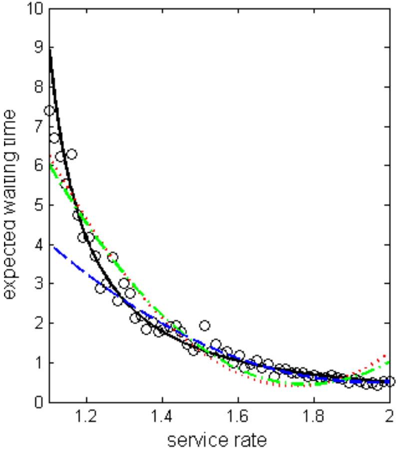
Regression Metamodeling?

Assumptions:

- $\mu(\theta) = b(\theta)\beta$ for known b and some β.
- All $\varepsilon_j(\theta)$ are independent.
- OLS: $\varepsilon_j(\theta) \sim N(0, v)$.
- WLS: $\varepsilon_j(\theta) \sim N(0, v(\theta))$.

Legend: (quadratic metamodels)

- black line = truth, $\circ$ = data
- fit to data: OLS, WLS
- red dots = best fit to truth



Misspecification: best fit may be bad.

Challenge of handling noise: dangers of WLS and OLS.

Metamodeling: Nonparametric Regression!

Does $\mu = b\beta$, a linear combination of basis functions?

Metamodeling: Nonparametric Regression!

Does $\mu = b\beta$, a linear combination of basis functions?

NO. The true form of μ is unknown.

Metamodeling: Nonparametric Regression!

Does $\mu = b\beta$, a linear combination of basis functions?

NO. The true form of μ is unknown.

Should we try to filter out the noise?

Metamodeling: Nonparametric Regression!

Does $\mu = b\beta$, a linear combination of basis functions?

NO. The true form of μ is unknown.

Should we try to filter out the noise?

YES. Monte Carlo produces noise.
Independence across simulation runs $\Rightarrow$ noise can be filtered.
Heteroscedasticity $\Rightarrow$ filtering is nontrivial.
Variance of each run can be estimated using multiple replications.

Metamodeling: Nonparametric Regression!

Does $\mu = b\beta$, a linear combination of basis functions?

NO. The true form of μ is unknown.

Should we try to filter out the noise?

YES. Monte Carlo produces noise.
Independence across simulation runs $\Rightarrow$ noise can be filtered.
Heteroscedasticity $\Rightarrow$ filtering is nontrivial.
Variance of each run can be estimated using multiple replications.

Approach: nonparametric regression

- smoothing splines
- kernel smoothing
- moving least squares (local regression)

Challenges: μ non-differentiable, discontinuous, high-dimensional

Stochastic Kriging

Simulation output at θ_i is $Y(\theta_i; n_i) = \mu(\theta_i) + \frac{1}{n_i} \sum_{j=1}^{n_i} \varepsilon_j(\theta_i)$.

Assumptions: (smoothing splines family)

- $\varepsilon_j(\theta) \sim N(0, v_j)$, "intrinsic" variance, all are independent.
- μ is a random field
  - $\mu(\theta)$ is normal with mean $b(\theta)\beta$.
  - $\mu(\theta)$ and $\mu(\theta')$ have "extrinsic" covariance $\sigma^2(\theta, \theta')$, data-driven spatial correlation modeling.

Stochastic Kriging

Simulation output at θ_i is $Y(\theta_i; n_i) = \mu(\theta_i) + \frac{1}{n_i}\sum_{j=1}^{n_i} \varepsilon_j(\theta_i)$.

Assumptions: (smoothing splines family)

- $\varepsilon_j(\theta) \sim N(0, v_j)$, "intrinsic" variance, all are independent.
- μ is a random field
  - $\mu(\theta)$ is normal with mean $b(\theta)\beta$.
  - $\mu(\theta)$ and $\mu(\theta')$ have "extrinsic" covariance $\sigma^2(\theta, \theta')$, data-driven spatial correlation modeling.

Prediction at θ given data $Y = [Y(\theta_1; n_1), \ldots, Y(\theta_k; n_k)]^\top$ is

$$\hat{\mu}(\theta) = b(\theta)\beta + w(\theta)(Y - B\beta),$$

where $Y - B\beta =$ residuals at design points.

Behavior: between regression and interpolation.

http://stochastickriging.net

DataBase Monte Carlo (DBMC)

DBMC vs. metamodeling

- Both store info from simulation not at θ to learn about $\mu(\theta)$.

DataBase Monte Carlo (DBMC)

DBMC vs. metamodeling

- Both store info from simulation not at θ to learn about $\mu(\theta)$.
- Does estimating $\mu(\theta)$ require a simulation at θ? DBMC—yes, metamodeling—no
- Is $\hat{\mu}(\theta)$ biased? DBMC—no, metamodeling—yes

DataBase Monte Carlo (DBMC)

DBMC vs. metamodeling

- Both store info from simulation not at θ to learn about $\mu(\theta)$.
- Does estimating $\mu(\theta)$ require a simulation at θ?
  DBMC—yes, metamodeling—no
- Is $\hat{\mu}(\theta)$ biased? DBMC—no, metamodeling—yes
- DBMC exploits structure of $Y(\omega, \cdot)$ vs. $\mu(\cdot) = E[Y(\omega, \cdot)]$

DBMC strategy

1. simulation run of N replications at θ_0 to generate <u>database</u> $(\omega_1, Y(\omega_1, \theta_0)), \ldots, (\omega_N, Y(\omega_N, \theta_0))$
2. use database to do variance reduction while simulating $n \ll N$ replications at θ to approximate $\mu(\theta)$

Due to Pirooz Vakili et al. Overview in:

`http://users.iems.northwestern.edu/~staum/MonteCarloFinance.pdf`

Motivation: If for θ near θ_0, the payoff $Y(\omega, \theta)$ is highly correlated with $Y(\omega, \theta_0)$, good control variate.

DBMC with Control Variates

Motivation: If for θ near θ_0, the payoff $Y(\omega, \theta)$ is highly correlated with $Y(\omega, \theta_0)$, good control variate.

Problem: $\mu(\theta_0) = E[Y(\omega, \theta_0)]$ is unknown.

DBMC with Control Variates

Motivation: If for θ near θ_0, the payoff $Y(\omega, \theta)$ is highly correlated with $Y(\omega, \theta_0)$, good control variate.

Problem: $\mu(\theta_0) = E[Y(\omega, \theta_0)]$ is unknown.

Solution: $Y(\omega, \theta_0)$ is a quasi-control variate.
Approximate $E[Y(\omega, \theta_0)]$ well using the database of large size N.
Sample $n \ll N$ random variates $u_1, \ldots, u_n$,

$$\hat{\mu}(\theta) = \frac{1}{n}\sum_{j=1}^{n} Y(u_j, \theta) - \beta\Big(\frac{1}{n}\sum_{j=1}^{n} Y(u_j, \theta_0) - \underbrace{\frac{1}{N}\sum_{j=1}^{N} Y(\omega_j, \theta_0)}_{\text{from database}}\Big).$$

http://www.informs-sim.org/wsc08papers/037.pdf

Structured Database Monte Carlo with Stratification

SDMC strategy

1. generate database $(\omega_1, Y(\omega_1, \theta_0)), \dots, (\omega_N, Y(\omega_N, \theta_0))$
2. structure the database, e.g., by sorting on $Y(\omega, \theta_0)$
3. use database to do variance reduction at θ

Structured Database Monte Carlo with Stratification

SDMC strategy

1. generate database $(\omega_1, Y(\omega_1, \theta_0)), \ldots, (\omega_N, Y(\omega_N, \theta_0))$
2. structure the database, e.g., by sorting on $Y(\omega, \theta_0)$
3. use database to do variance reduction at θ

Stratification after sorting

1. partition $\{\omega_1, \ldots, \omega_N\}$ into n contiguous strata
2. stratified resampling of $u_1, \ldots, u_n$ from $\{\omega_1, \ldots, \omega_N\}$
3. $\hat{\mu}(\theta) = \sum_{j=1}^{n} Y(u_j, \theta)/n$ if strata are size N/n.

Structured Database Monte Carlo with Stratification

SDMC strategy

1. generate database $(\omega_1, Y(\omega_1, \theta_0)), \ldots, (\omega_N, Y(\omega_N, \theta_0))$
2. structure the database, e.g., by sorting on $Y(\omega, \theta_0)$
3. use database to do variance reduction at θ

Stratification after sorting

1. partition $\{\omega_1, \ldots, \omega_N\}$ into n contiguous strata
2. stratified resampling of $u_1, \ldots, u_n$ from $\{\omega_1, \ldots, \omega_N\}$
3. $\hat{\mu}(\theta) = \sum_{j=1}^{n} Y(u_j, \theta)/n$ if strata are size N/n.

Advantages vs. manual stratification of hypercube:

- don't need stratum probabilities or conditional sampling
- automatically creates good strata if $Y(\cdot, \theta)$ and $Y(\cdot, \theta_0)$ are nearly comonotone

`http://www.informs-sim.org/wsc08papers/036.pdf` and references therein

Simulation on Demand for Pricing Many Securities

Goal: accurate approximation of $\mu(\theta, \psi_1), \ldots, \mu(\theta, \psi_J)$
where θ = market scenario, ψ_j = security j parameters.

Stochastic kriging (SK) metamodels $\hat{\mu}(\cdot, \psi_j)$

1. Establish likely region Θ for future scenarios.
2. Simulate accurately at $\theta_1, \ldots, \theta_k$ chosen to fill Θ.
3. Build and cross-validate SK metamodels $\hat{\mu}(\cdot, \psi_1), \ldots, \hat{\mu}(\cdot, \psi_J)$.
4. If they don't pass, add more scenarios or simulation effort, return to Step 2.

Simulation on Demand for Pricing Many Securities

Goal: accurate approximation of $\mu(\theta, \psi_1), \ldots, \mu(\theta, \psi_J)$
where θ = market scenario, ψ_j = security j parameters.

Stochastic kriging (SK) metamodels $\hat{\mu}(\cdot, \psi_j)$

1. Establish likely region Θ for future scenarios.
2. Simulate accurately at $\theta_1, \ldots, \theta_k$ chosen to fill Θ.
3. Build and cross-validate SK metamodels $\hat{\mu}(\cdot, \psi_1), \ldots, \hat{\mu}(\cdot, \psi_J)$.
4. If they don't pass, add more scenarios or simulation effort, return to Step 2.

Result: After 2.2 hours on one PC for $J = 75$ securities, all root average relative MSEs were $< 0.75\%$.

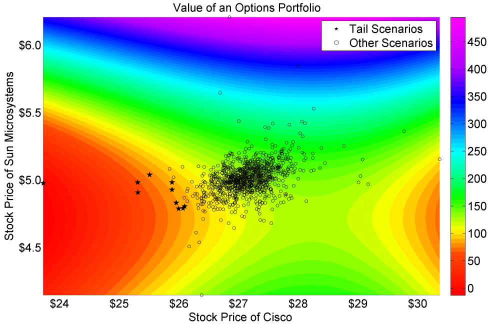
Easy to parallelize.

`http://users.iems.northwestern.edu/~nelsonb/SK/valuation.pdf`

Expected Shortfall with Stochastic Kriging

Nested simulation of expected shortfall = CVaR at level p

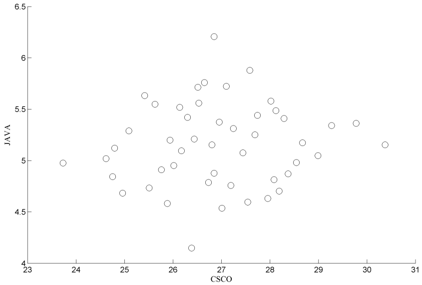
1. Simulate scenarios $\Theta_1, \ldots, \Theta_K$.
2. In each scenario, approx. portfolio value $\mu(\Theta_i)$ by $\hat{\mu}(\Theta_i)$.
3. Choose $V_1, \ldots, V_{Kp}$ to be lowest values in $\hat{\mu}(\Theta_1), \ldots, \hat{\mu}(\Theta_K)$.
4. Approx. ES by $-\sum_{i=1}^{Kp} V_i$. (Only tail scenarios matter.)



Expected Shortfall with Stochastic Kriging

Stochastic kriging (SK) for portfolio valuation

1. Simulate at scenarios chosen to fill space.

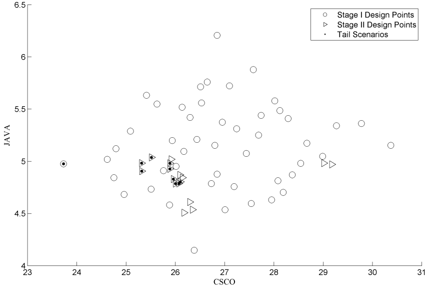


http://users.iems.northwestern.edu/~staum/skes.pdf

Expected Shortfall with Stochastic Kriging

Stochastic kriging (SK) for portfolio valuation

1. Simulate at scenarios chosen to fill space.
2. Use SK to choose scenarios likeliest to be in the tail, simulate at them.

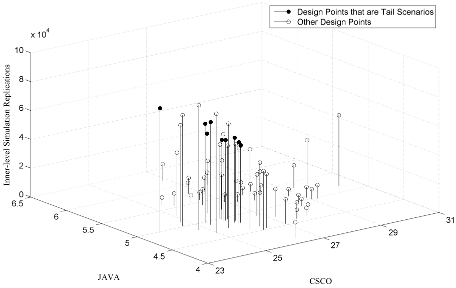


http://users.iems.northwestern.edu/~staum/skes.pdf

Expected Shortfall with Stochastic Kriging

Stochastic kriging (SK) for portfolio valuation

1. Simulate at scenarios chosen to fill space.
2. Use SK to choose scenarios likeliest to be in the tail, simulate at them.
3. Increase replications.

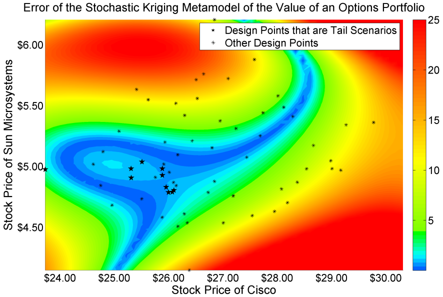


http://users.iems.northwestern.edu/~staum/skes.pdf

Expected Shortfall with Stochastic Kriging

Stochastic kriging (SK) for portfolio valuation

1. Simulate at scenarios chosen to fill space.
2. Use SK to choose scenarios likeliest to be in the tail, simulate at them.
3. Increase replications.
4. Final SK metamodel.

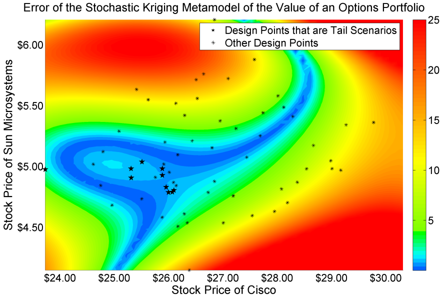


http://users.iems.northwestern.edu/~staum/skes.pdf

Expected Shortfall with Stochastic Kriging

Stochastic kriging (SK) for portfolio valuation

1. Simulate at scenarios chosen to fill space.
2. Use SK to choose scenarios likeliest to be in the tail, simulate at them.
3. Increase replications.
4. Final SK metamodel.



Result: RMSE 50 times better than standard nested simulation

`http://users.iems.northwestern.edu/~staum/skes.pdf`

Some Future Applied Research

Nested simulation:

apply metamodeling (Hong&Juneja; Liu&Staum) or DBMC

- adaptive allocation (Broadie,Du&Moallemi; Gordy&Juneja; Liu&Staum)
- achieve rate of MSE convergence closer to $1/C$?

Some Future Applied Research

Nested simulation:

apply metamodeling (Hong&Juneja; Liu&Staum) or DBMC

- adaptive allocation (Broadie,Du&Moallemi; Gordy&Juneja; Liu&Staum)
- achieve rate of MSE convergence closer to $1/C$?

American options: what nonparametric regression ideas to use?

(Carrière; Tompaidis&Yang)

- reduce the need for problem-specific basis functions
- tailor to the yes/no objective

Some Future Applied Research

Nested simulation:
apply metamodeling (Hong&Juneja; Liu&Staum) or DBMC

- adaptive allocation (Broadie,Du&Moallemi; Gordy&Juneja; Liu&Staum)
- achieve rate of MSE convergence closer to $1/C$?

American options: what nonparametric regression ideas to use?
(Carrière; Tompaidis&Yang)

- reduce the need for problem-specific basis functions
- tailor to the yes/no objective

In general: algorithm design
for computational efficiency, validation, updating

Some Future Methodological Research

Metamodeling:

- When to use what nonparametric regression techniques?
- Estimate variance at each θ from multiple replications? How to use these estimates?
- Analyze and reduce bias.

Some Future Methodological Research

Metamodeling:

- When to use what nonparametric regression techniques?
- Estimate variance at each θ from multiple replications? How to use these estimates?
- Analyze and reduce bias.

Experiment design:

place design points, allocate effort, manage noise-to-signal ratio

- Metamodeling: lots known from statistics, more to do for simulation (adaptive; many replications)
- DBMC: open

Some Future Methodological Research

Metamodeling:

- When to use what nonparametric regression techniques?
- Estimate variance at each θ from multiple replications? How to use these estimates?
- Analyze and reduce bias.

Experiment design:

place design points, allocate effort, manage noise-to-signal ratio

- Metamodeling: lots known from statistics, more to do for simulation (adaptive; many replications)
- DBMC: open

Combining metamodeling and DBMC

A Green Vision for Financial Computing

When introducing a model:

1. Establish a domain Θ of parameters (e.g. strike, vol)
2. Run simulations until metamodel $\hat{\mu}$ is good in domain Θ.

A Green Vision for Financial Computing

When introducing a model:

1. Establish a domain Θ of parameters (e.g. strike, vol)
2. Run simulations until metamodel $\hat{\mu}$ is good in domain Θ.

When using a model at θ:

- Expand domain if $\theta \notin \Theta$.
- For low-fidelity applications, use the metamodel: $\hat{\mu}(\theta)$.
- For high-fidelity applications,
  1. Run more simulations at θ until metamodel is good at θ.
  2. Update the metamodel.

A Green Vision for Financial Computing

When introducing a model:

1. Establish a domain Θ of parameters (e.g. strike, vol)
2. Run simulations until metamodel $\hat{\mu}$ is good in domain Θ.

When using a model at θ:

- Expand domain if $\theta \notin \Theta$.
- For low-fidelity applications, use the metamodel: $\hat{\mu}(\theta)$.
- For high-fidelity applications,
  1. Run more simulations at θ until metamodel is good at θ.
  2. Update the metamodel.

Efficient computing: reduce and *reuse!*

- Store output (possibly condensed) of every simulation run in the metamodel or the database(s).
- Discard output data only when model is abandoned.