Sequential Regression Methods for Adaptive Control

Mike Ludkovski

Dept of Statistics & Applied Probability UC Santa Barbara

FiME Seminar, IHP, April 11, 2014
Work supported by NSF DMS-1222262
Joint with Bobby Gramacy (Chicago), Jarad Niemi (Iowa State), Ekaterina Shatskikh, Ruimeng Hu (all UCSB)

Main Message

Monte Carlo for Optimal Stopping = Regression + Stochastic Grid

Main Message

Monte Carlo for Optimal Stopping = Regression +

Main Message

Monte Carlo for Optimal Stopping = Regression + Stochastic Grid

- Adapt the grids to find the stopping boundary
- Crash course in sequential design

Motivation I: Pricing Bermudan Contracts

- Financial contract with payoff $h(X)$ on underlying (X_t) (eg. asset price)
- Can exercise at any time $\tau = 1, 2, \ldots, T$. Decision must be based on $X_{1:\tau}$.
- Fair value of contract is $V(t,x) = \sup_\tau \mathbb{E}^Q[h(X_\tau)|X_t = x]$
- Interested in the price $V(t,x)$ as well as exercise strategy τ^*
- If (X_t) is Markov, τ^* is the first hitting time of the stopping set $\mathfrak{S} = \{(t,x) : V(t,x) = h(x)\}$

Motivation II: Gas storage

- Optimally inject/withdraw gas over $[0, T]$ to maximize profit
- State variables: gas price(s) + inventory
- Action is { inject, withdraw, store }

Motivation III: Mitigating Flu Epidemics

- Flu outbreaks occur every winter. Timing/severity vary enormously year-to-year.
- Goal is policy-making: how to efficiently respond to an outbreak given information so far
- eg: every week choose among "Do Nothing", "Vaccinate", "Quarantine", "Get Better Data". (see L.–Lin (2013))

Motivation IV: Target tracking

- Engineers aim to seq control multiple UAV drones for tracking
- Stochastic noise (wind, target evasion, etc.) is important
- Drone has constant speed/height. Control the discrete banking angle

Sequential Control

- The above examples are special cases of adaptive dynamic controls
- Actions must be done sequentially
- Underlying models are **stochastic**, have complex nonlinear dynamics and multi-dimensional state spaces
- Action space is small (discrete: $|\mathcal{A}| < 10$)
- Optimal behavior is sought

Basic Setting: Optimal Stopping on Finite Horizon

- Stopping regions indexed by time: $\mathfrak{S}_t = \{x : V(t,x) = h(x)\}$: $\tau^* = \inf\{t : X_t \in \mathfrak{S}_t\} \wedge T$
- Backwards Induction: initialize $\mathfrak{S}_T = \{x : h(x) \geq 0\}$
- Timing value:

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

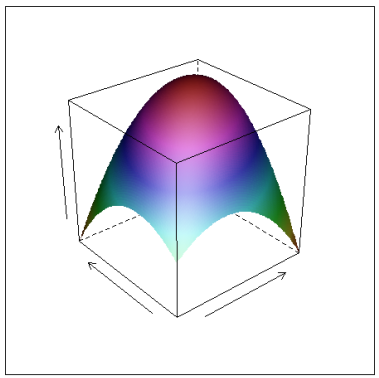
- Then $\mathfrak{S}_t = \{x : T_t(x) < 0\}$
- To find τ^*, it's sufficient to evaluate the conditional expectation, i.e. approximate $T_t(x)$
- The latter sub-problem is well-suited for Monte Carlo approaches

Conditional Expectations via Monte Carlo

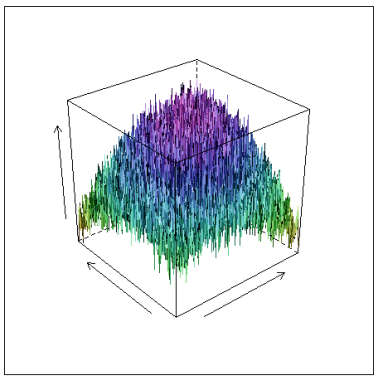
- Given: $\hat{\mathfrak{S}}_{t+1:T}$ and starting point X_t
- Simulate trajectory $X_{t+1:T}$ and take $\tau' = \inf\{s > t : X_s \in \hat{\mathfrak{S}}_s\}$
- Then $y_t := h(X_{\tau'}) - h(X_t)$ is a realization of the r.v. inside $T_t(x)$
- i.e. $\mathbb{E}[y_t] = T_t(x)$ and so can do a vanilla MC (average many y's) to estimate $T_t(x)$

Conditional Expectations via Monte Carlo

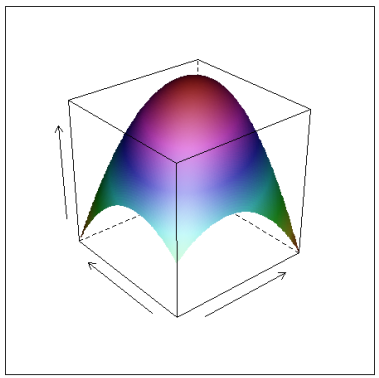
- Given: $\hat{\mathfrak{S}}_{t+1:T}$ and starting point X_t
- Simulate trajectory $X_{t+1:T}$ and take $\tau' = \inf\{s > t : X_s \in \hat{\mathfrak{S}}_s\}$
- Then $y_t := h(X_{\tau'}) - h(X_t)$ is a realization of the r.v. inside $T_t(x)$
- i.e. $\mathbb{E}[y_t] = T_t(x)$ and so can do a vanilla MC (average many y's) to estimate $T_t(x)$
- Generate a stochastic grid $(x_t^n)_{n=1}^N$ and paths $x_{t+1:T}^{1:N}$
- Obtain a sample $\{x_t, y_t\}^{1:N}$
- Perform regression to estimate $\hat{T}_t(x)$ and hence $\hat{\mathfrak{S}}_t$



Truth



Simulation



Regression

Regression Monte Carlo (RMC) Approach (aka Longstaff-Schwartz)

- Focus on conditional expectations (Snell envelope methods) – build up the sets $\mathfrak{S}_{1:T}$ iteratively
- Key idea: approximate $\mathfrak{S}_t$ – only care about the sign of $T_t(x)$
- **Accuracy is solely driven by the accuracy of the stopping sets $\hat{\mathfrak{S}}_t$**

Regression Monte Carlo (RMC) Approach (aka Longstaff-Schwartz)

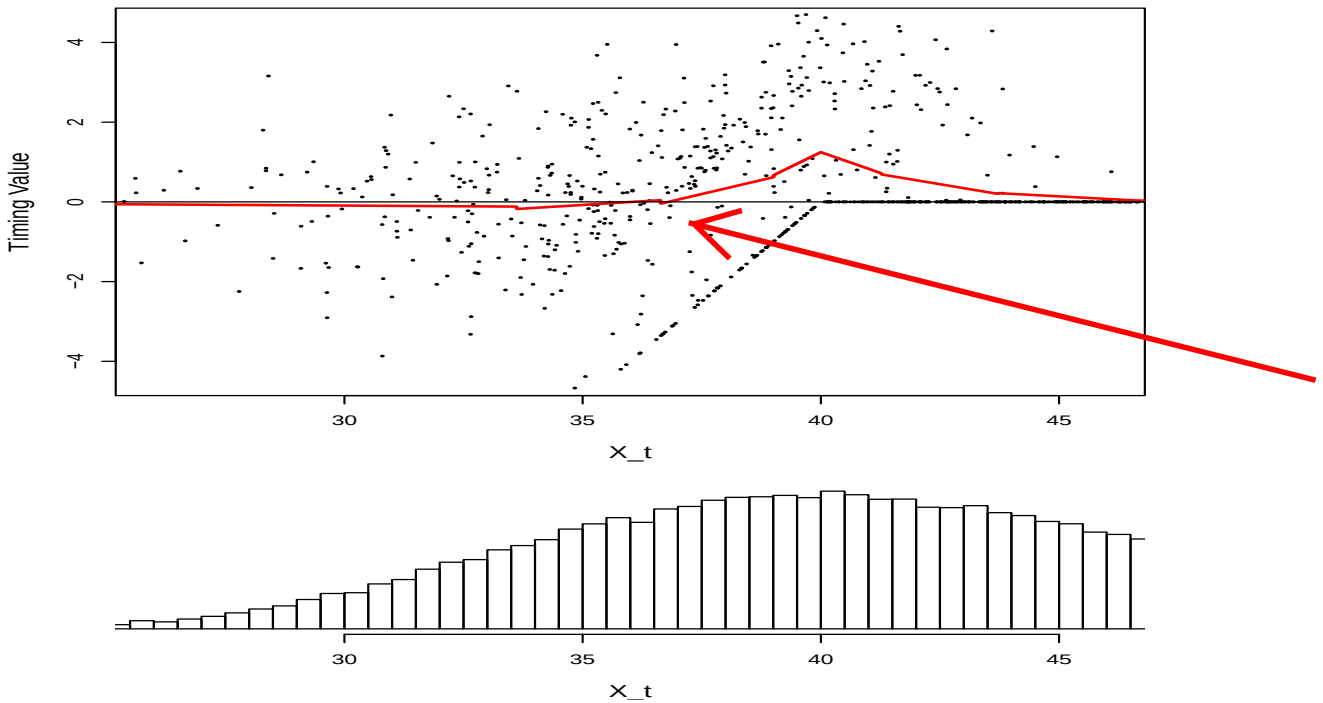
- Focus on conditional expectations (Snell envelope methods) – build up the sets $\mathfrak{S}_{1:T}$ iteratively
- Key idea: approximate $\mathfrak{S}_t$ – only care about the sign of $T_t(x)$
- **Accuracy is solely driven by the accuracy of the stopping sets** $\hat{\mathfrak{S}}_t$
- Value function V is never used explicitly
- Given the stopping strategy $\hat{\mathfrak{S}}_{0:T}$ compute

$$\hat{V}(0,x) = \frac{1}{M}\sum_{m=1}^{M} h(X^m_{\tau_m}), \qquad \tau_m = \min\{s : X^m_s \in \hat{\mathfrak{S}}_s\}$$

- For M large enough, $\hat{V}$ is a guaranteed lower-bound ($\hat{\mathfrak{S}}_t$ is suboptimal)

Vanilla RMC

- Standard approach is to generate the grids via $x_t^n \sim p(X_t|X_0)$ i.i.d., (a fixed forward trajectory database)
- For regression LSMC used OLS with bases $\{B_r(x)\}$ [L^2 projection]
- Observe the *wild* distribution of (x_t, y_t) samples below



RMC Strengths and Weaknesses

- RMC allows to deal with arbitrarily complex dynamics in (X_t)
- RMC identifies the benefit of the simple action space – policy search vs value function approx.
- *First real success of Monte Carlo methods in control* – 1900+ citations and counting

Two key parts of RMC are the stochastic grids (x_t^n) and the regression method used for estimating $T_t(x)$

RMC Strengths and Weaknesses

- RMC allows to deal with arbitrarily complex dynamics in (X_t)
- RMC identifies the benefit of the simple action space – policy search vs value function approx.
- *First real success of Monte Carlo methods in control* – 1900+ citations and counting

Two key parts of RMC are the stochastic grids (x_t^n) and the regression method used for estimating $T_t(x)$

- Inefficient: many x_t^n's might be far from where the action is.

RMC Strengths and Weaknesses

- RMC allows to deal with arbitrarily complex dynamics in (X_t)
- RMC identifies the benefit of the simple action space – policy search vs value function approx.
- *First real success of Monte Carlo methods in control* – 1900+ citations and counting

Two key parts of RMC are the stochastic grids (x_t^n) and the regression method used for estimating $T_t(x)$

- Inefficient: many x_t^n's might be far from where the action is.
- Unstable: OLS is sensitive to outliers, noise, etc.

RMC Strengths and Weaknesses

- RMC allows to deal with arbitrarily complex dynamics in (X_t)
- RMC identifies the benefit of the simple action space – policy search vs value function approx.
- *First real success of Monte Carlo methods in control* – 1900+ citations and counting

Two key parts of RMC are the stochastic grids (x_t^n) and the regression method used for estimating $T_t(x)$

- Inefficient: many x_t^n's might be far from where the action is.
- Unstable: OLS is sensitive to outliers, noise, etc.
- Dependent: extra correlation between $\hat{\mathfrak{S}}_t$ and $\hat{\mathfrak{S}}_{t+1}$

RMC Strengths and Weaknesses

- RMC allows to deal with arbitrarily complex dynamics in (X_t)
- RMC identifies the benefit of the simple action space – policy search vs value function approx.
- *First real success of Monte Carlo methods in control* – 1900+ citations and counting

Two key parts of RMC are the stochastic grids (x_t^n) and the regression method used for estimating $T_t(x)$

- Inefficient: many x_t^n's might be far from where the action is.
- Unstable: OLS is sensitive to outliers, noise, etc.
- Dependent: extra correlation between $\hat{\mathfrak{S}}_t$ and $\hat{\mathfrak{S}}_{t+1}$
- No auto-pilot: how to pick the bases in multiple dimensions
- Approximation quality is bounded *a priori* by the bases picked

RMC Strengths and Weaknesses

- RMC allows to deal with arbitrarily complex dynamics in (X_t)
- RMC identifies the benefit of the simple action space – policy search vs value function approx.
- *First real success of Monte Carlo methods in control* – 1900+ citations and counting

Two key parts of RMC are the stochastic grids (x_t^n) and the regression method used for estimating $T_t(x)$

- Inefficient: many x_t^n's might be far from where the action is.
- Unstable: OLS is sensitive to outliers, noise, etc.
- Dependent: extra correlation between $\hat{\mathfrak{S}}_t$ and $\hat{\mathfrak{S}}_{t+1}$
- No auto-pilot: how to pick the bases in multiple dimensions
- Approximation quality is bounded *a priori* by the bases picked
- No consideration of computational budget: **"generate as many paths as you can"**

Current state-of-the-art

- Empirically it is well known that OLS is unstable. Many alternatives have been tried: Bouchard and Warin (2011), Kohler (2011), Stentoft (2012), Tompaidis and Yang (2013) ,...
- Other simulation methods rely on clever stochastic meshes: Broadie and Glasserman (2001), Hu et al. (2012)
- Ongoing theoretical developments focusing on error propagation through dependent regressions: Egloff (2004), Gobet and Warin (2006), Belomestny (2011), Kohler (2012), Zanger (2013) — mostly for methods that directly approximate $V(t, x)$
- Numerous applications beyond American options (eg L. & Niemi 2012 for control of epidemics) where computational issues become especially acute

New Approach

- Face an iterative statistical modeling problem
- Unknown $x \mapsto T_t(x)$ modeled by $x \mapsto f_t(x)$
- Loss function is $\mathbb{E}_{0,x_0}\left[|T_t(X_t^*)|1_{\{\text{sign } T_t(X_t^*) \neq \text{sign } f_t(X_t^*)\}}\right]$
- i.e. : find the zero-contour of $T_t(x)$ (aka $\partial\mathfrak{S}_i(t)$ boundary of the stopping set)
- Do NOT need uniform accuracy on the whole state space
- Can localize modeling effort and therefore improve efficiency
- In the spirit of stochastic global optimization

Reformulate as a sequential design problem: given a generating model (x, y) where $y = T_t(x) + \varepsilon$, efficiently approximate the classifier $C(x) = 1_{T(t,x)>0}$.

Old vs. New approaches to RMC

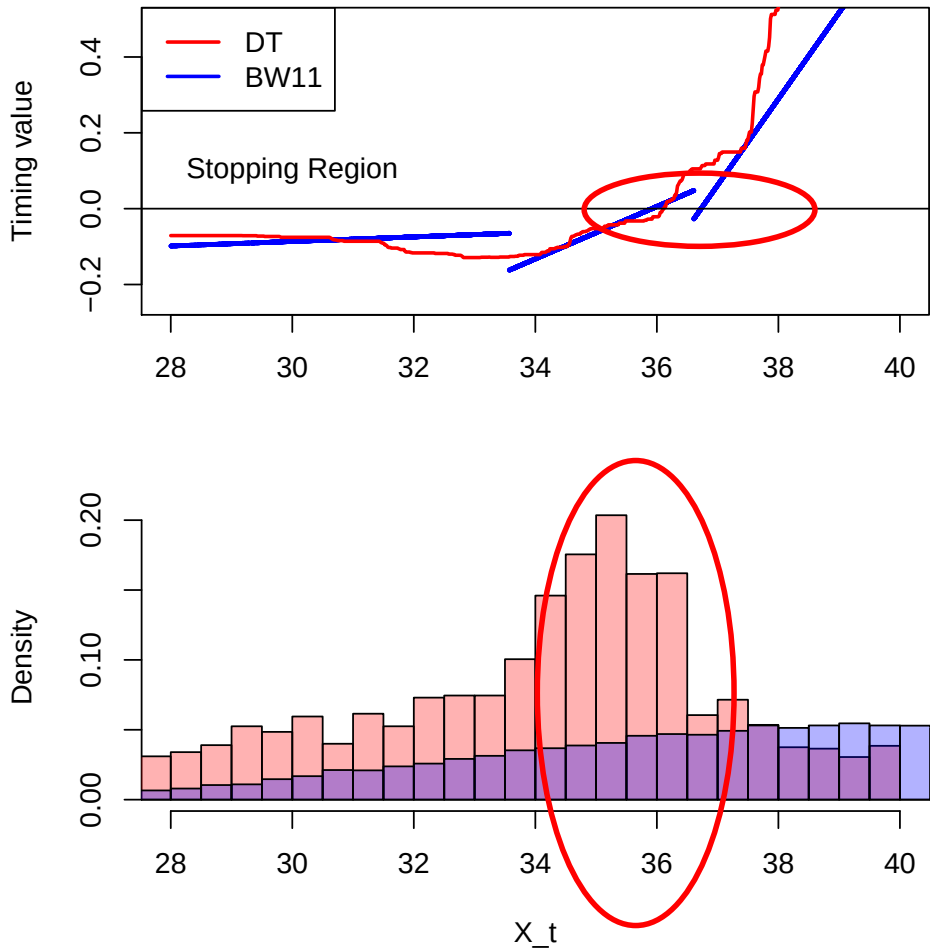


Figure : Old vs. new approaches to RMC

Dynamic Regression Monte Carlo

Sample the grids adaptively online, to localize fitting domain.

- Start with initial grid $\{x_t^{(0)}\}$
- Build initial approximation $\mathfrak{S}_t^{(0)}$
- LOOP for $k = 0, 1, \ldots$
  - Identify regions where the fit is uncertain/boundary $\partial\mathfrak{S}_t^{(k)}$ is located
  - Generate new paths $\{x_{t:T}^{(k),n'}\}$ and corresponding costs-to-go $y_t^{(k),n'} = h(x_{\tau^{n'}}^{(k),n'}) - h(x_t^{(k),n'})$.
  - Update the fit to $\mathfrak{S}_t^{(k+1)}$
- END LOOP
- Repeat above at each time-step $t = T - 1, \ldots, 1$

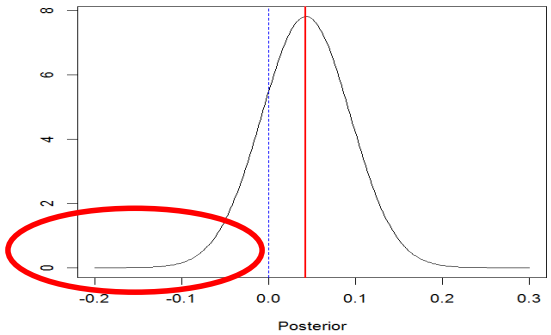
Can reduce the number of simulated paths by an order of magnitude!

Fit Uncertainty

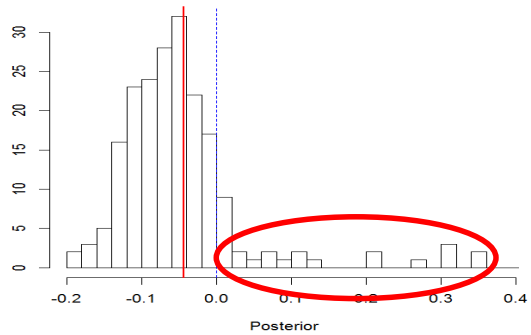
- Treat unknown $T_t(\cdot)$ as random function
- Given data $\mathcal{F}^n = \{(x_t^1, y_t^1), \ldots (x_t^n, y_t^n)\}$, have a posterior distribution $f_t|\mathcal{F}^n$
- Posterior mean $\hat{f}^n(x)$ is typically the estimator of $T_t(x)$, posterior variance $Var(T_t(x)|\mathcal{F}^n)$ is related to standard error at x.
- The accuracy at location x is

$$L^n(x) = \mathbb{E}\left[|T_t(x)|1_{\{\text{sign } T_t(x) \neq \text{sign } \hat{f}_t^n(x)\}}\right],$$

related to a quantile of the distrib of $f_t(x)|\mathcal{F}^n$



Gaussian loss



Empirical loss

Sequential Design

- Overall accuracy is $\mathcal{L}(f;(x,y)^{1:n}) = \int L^n(x)\mu_{X_t^*}(dx)$
- Choose new point x^{n+1} to maximize expected accuracy gain over next iteration:

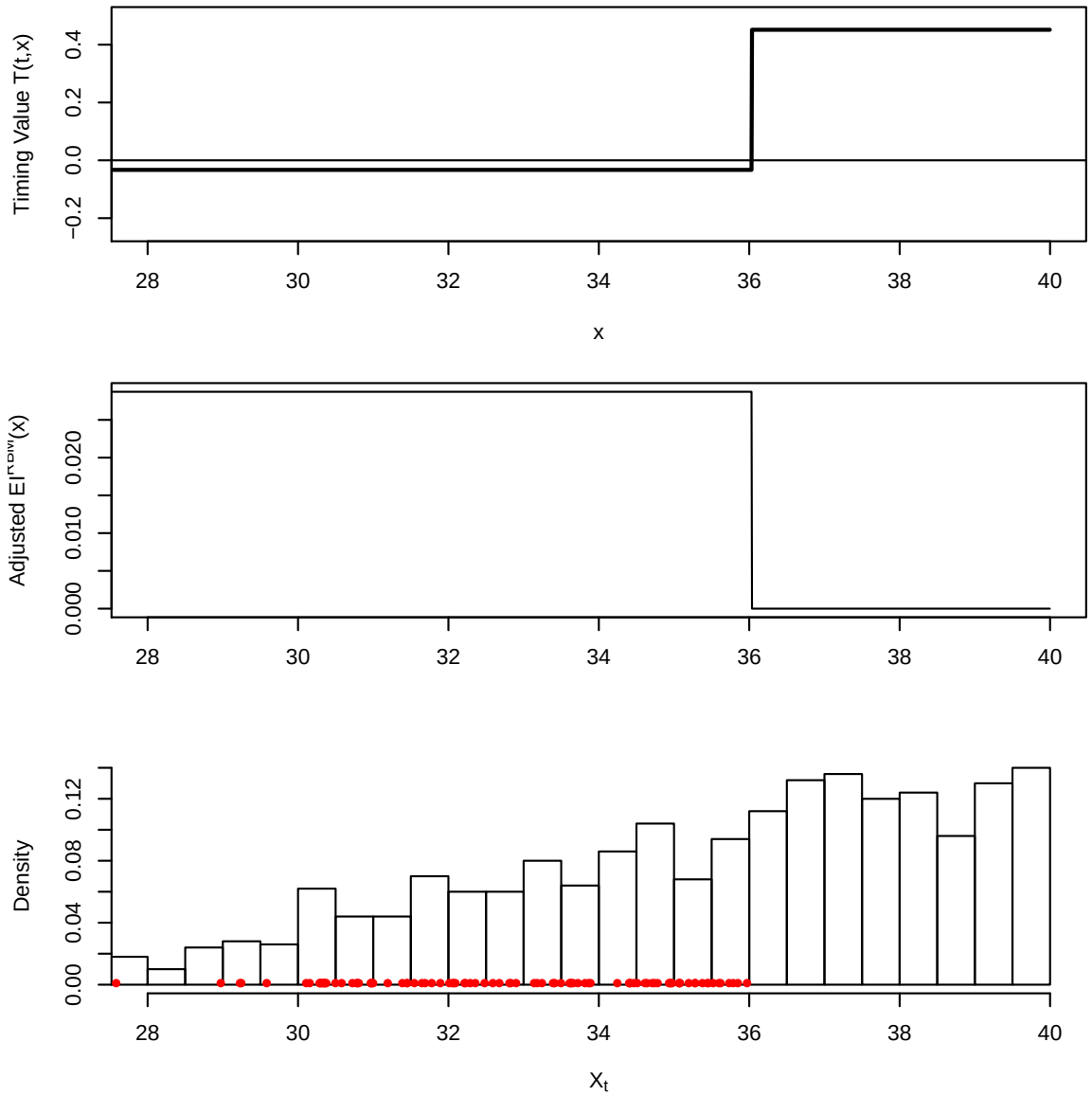
$$\sup_{x\in\mathcal{X}} \mathbb{E}[\mathcal{L}(f;(x,y)^{1:n} \cup (x^{n+1}, Y^{n+1})] - \mathcal{L}(f;(x,y)^{1:n})$$

- The above requires
  - integrating out the predictive distribution of $Y^{n+1}(x)|\mathcal{F}^n$
  - computing the effect of an *extra data sample* on $f|\mathcal{F}^{n+1}$.
  - optimizing over all locations x
  - Integrating out wrt law of X_t^*
- All difficult problems. Instead:
  - Use a simple heuristic to give score to points $I(x)$ (eg $I^n(x) = L^n(x)p_{X_t}(x)$)
  - Pick candidate locations $(x_1', x_2', \ldots)$ (eg space-filling design)
  - Sample x^{n+1} from x_m' using a potential based on $I(x_m')$
  - Can do the above sampling in batches (pick multiple x')

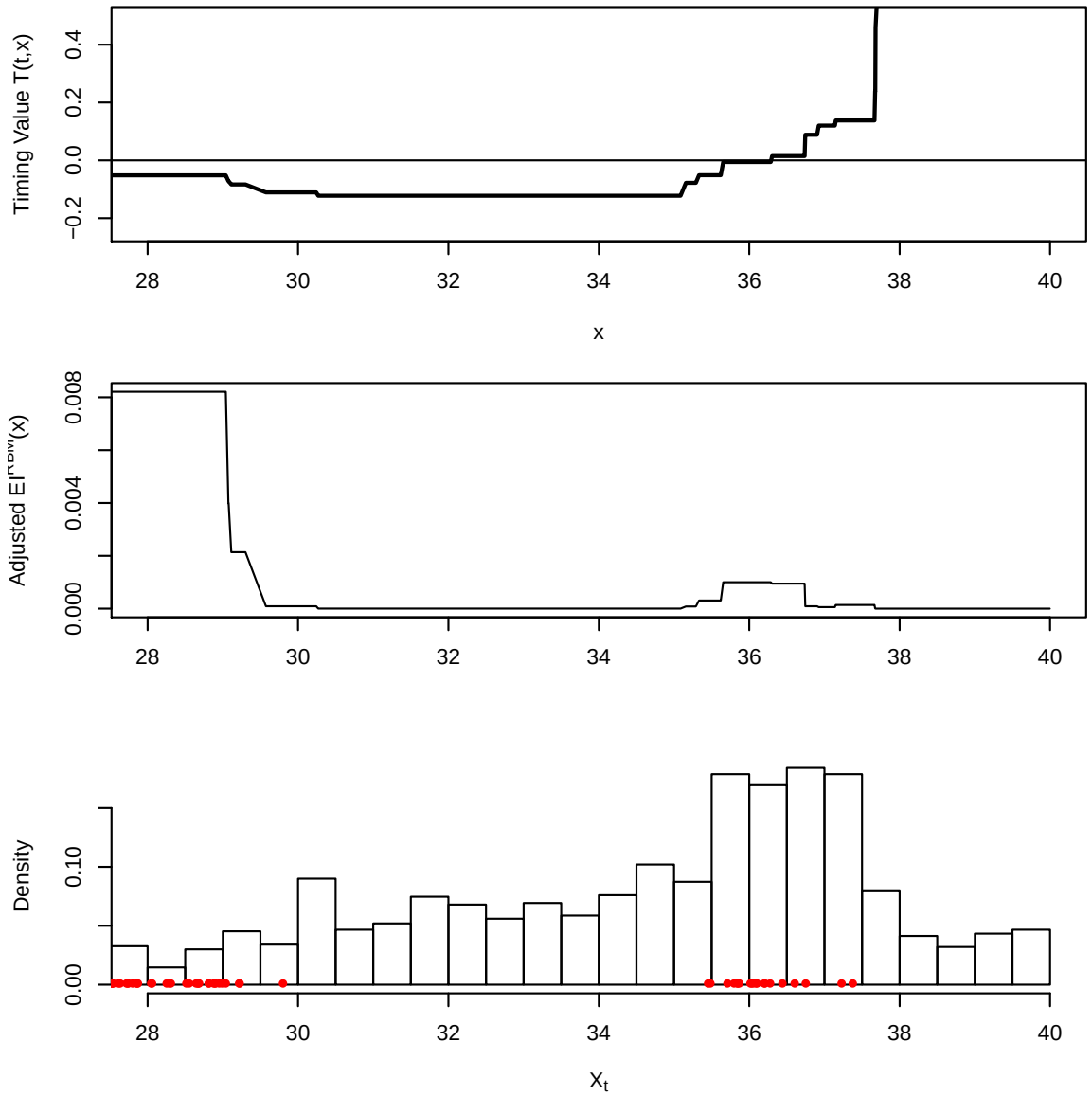
Classification MC?

- In fact do not need full $T_t(x)|\mathcal{F}^n$
- Only $\mathbb{P}(T_t(x) > 0|\mathcal{F}^n)$ – could use classification methods
- cf. Picazo (2002)
- Challenges: very low signal-to-noise ratio; loss function is not purely about classifying.

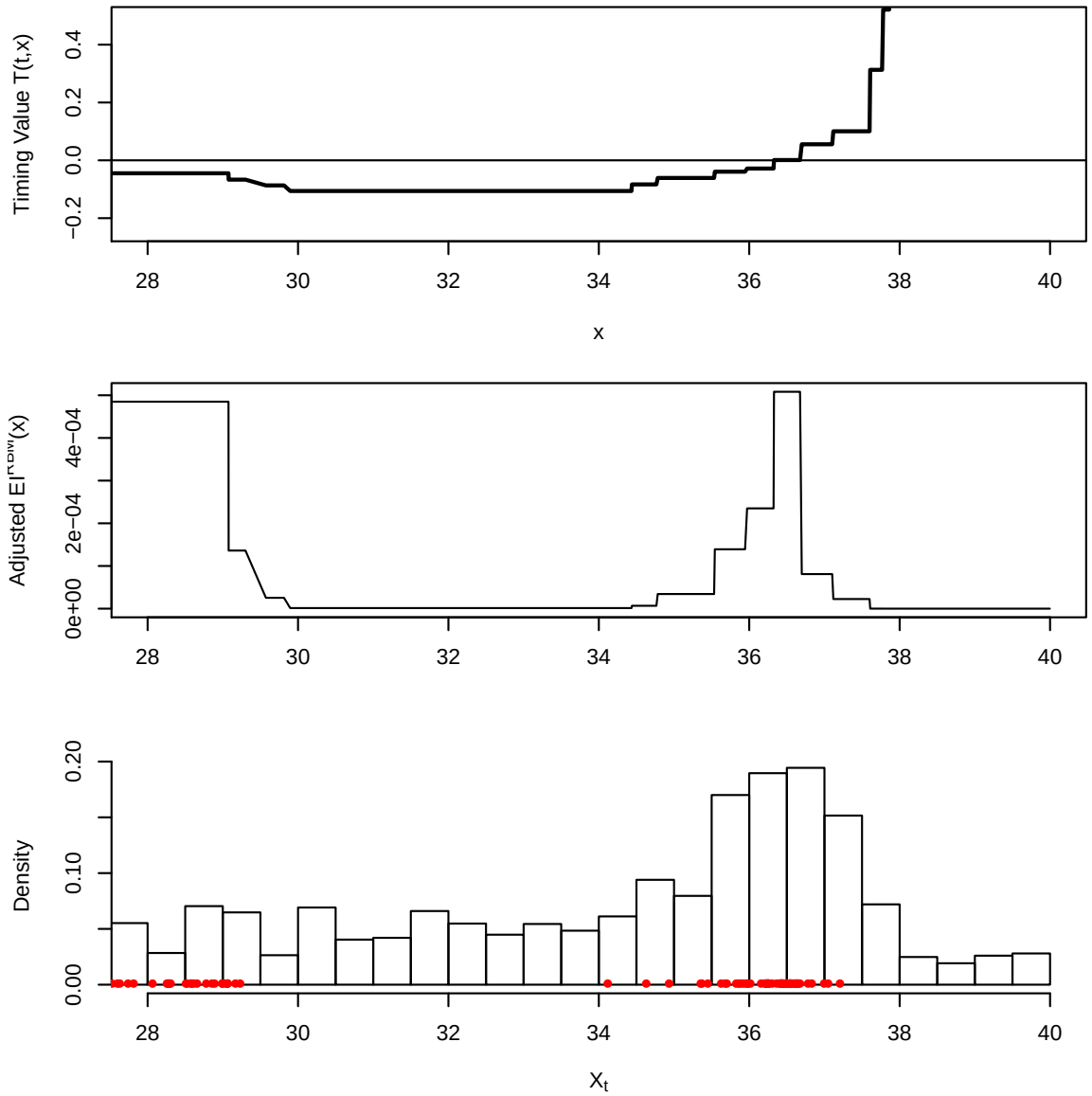
Active Learning: iteration $k = 0$



Active Learning: $k = 20$



Active Learning: $k = 40$



Statistical Aspects

- The fit must be updated after each grid refinement – computational bottleneck
- Also the regression approach must
  - include predictive uncertainty of the fit
  - deal with varying density of grid points
  - deal with wild error distributions: low signal-to-noise, skewed, multi-modal, heteroscedastic
- Strategies from recent developments in statistical response modeling:
  - data partitioning – local regression;
  - non-parametric approaches (complexity dynamically increases in n)
  - ensemble methods: combine many simple fits together
  - **Bayesian** point-of-view

Sequential Regression I: Dynamic Trees

- Developed by Gramacy, Polson and Taddy (2011)
- Bayesian treed nonparametric regression; `dynaTree` R package
- Model the fit as a collection of binary partition trees
- Works as an interacting particle system to grow the trees
- Provides full predictive distributions and has been used for active learning (Gramacy & Polson 2011)
- Fits are automatically locally refined as more grid points are added

Sequential Regression II: Stochastic Kriging

- Assume data comes from a Gaussian process
- Fitted $T_t(\vec{x})|\mathcal{F}^n$ is multivariate Gaussian
- Infinitely smooth
- Can be quickly updated (assuming fixed covariance structure)
- Variations exist to handle non-stationarity or heteroscedasticity (treed GP, SMC+GP, local GP, etc.)
- `tgp`, `plgp`, `DiceKrig`, R packages

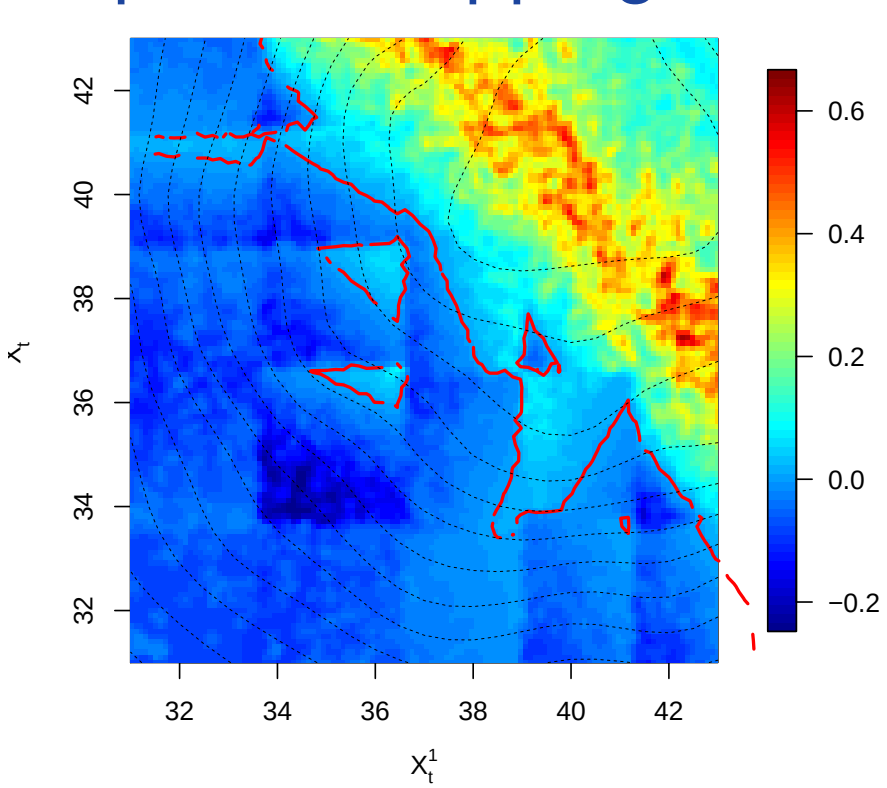
Implementation Issues

- Must balance exploration (to guarantee consistency as $n \to \infty$) and exploitation (fast zooming in toward stopping boundary)
- Sequential design is intermediate step – not so important to be accurate
- Use one regression method during active learning and another for final fit
- Regression = Smoothing + Interpolation (to do with noise/response smoothness)
- Take grid sizes N_t variable: more sample sites for t large to minimize error propagation
- Can add forward path values $x_{t+1:T}^{(k),n'}$ to the future grids
- Can input previous time-step $t+1$-fit as prior for the new fit at t

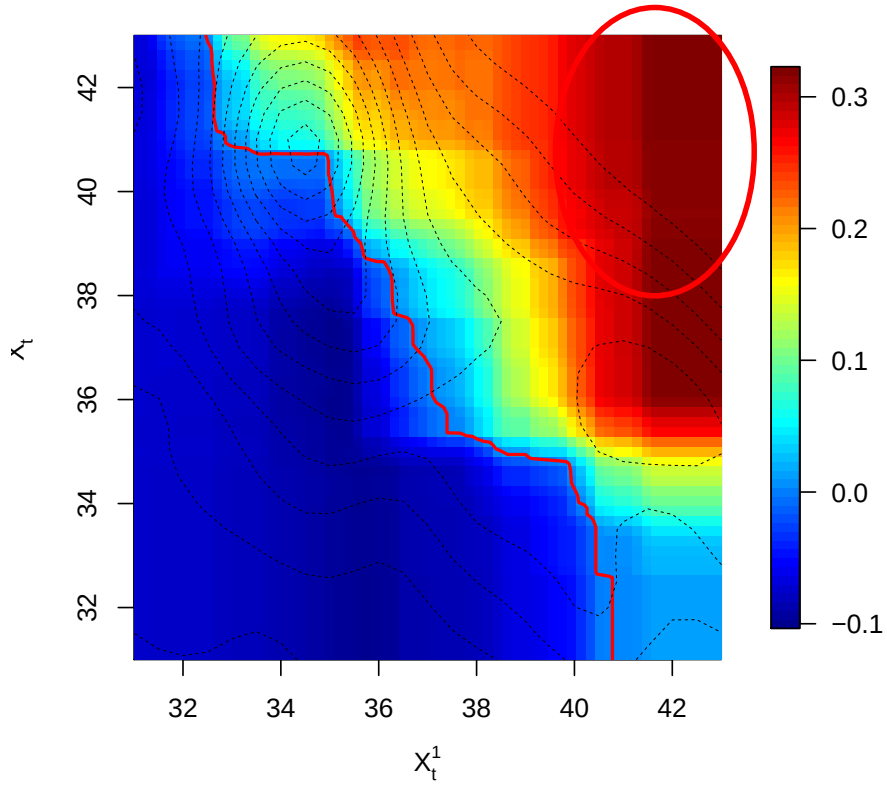
Require: N – number of initial grid points

$\mathfrak{S}_T \leftarrow \mathcal{X}$
for $t = T-1, T-2, \ldots, 0$ **do**
  $k \leftarrow 0$
  Generate an initial grid $\{x_t^{1:N}\}$, and corresponding classifier $\mathfrak{S}_t^{(0)}$
  while the current grid needs refining **do**
    $k \leftarrow k+1$
    Generate new grid point(s) $\{x_t^{(k),n'}\}$ $n' = 1, \ldots, N^{(k)}$
    Simulate forward trajectories $x_{t+1:T}^{(k),1:N^{(k)}}$. Using $\hat{\mathfrak{S}}_{t+1:T}$ find $y^{(k),1:N^{(k)}}$
    Update the classifier to $\mathfrak{S}_t^{(k)}$ using new samples $(x_t^{(k)}, y^{(k)})^{1:N^{(k)}}$
    (Update the classifiers $\hat{\mathfrak{S}}_{t+1:T-1}$ using $x_{t+1:T-1}^{(k),1:N^{(k)}}$)
    Save the overall grid $\{x_t\} \leftarrow \{x_t\} \cup \{x_t^{(k),1:N^{(k)}}\}$
  end while
  Generate final estimate of the classifier at time step t, $\hat{\mathfrak{S}}_t$
end for
Simulate forward trajectories $X_{0:T}^n$ from $X^n = x_0$ using $\hat{\mathfrak{S}}_{0:T}$
return $V(0, x_0) \simeq \frac{1}{N} \sum_{n=1}^{N} h_{\tau^n}(X_{\tau^n}^n)$
return Estimated policy $\{\hat{\mathfrak{S}}_{0:T}\}$.

2D Optimal Stopping Example



BW11



DT

Figure : Comparison of the Bouchard-Warin (left) and dynamic trees (right) fits in 2 dimensions. The heatmap indicates the levels of the estimated timing value $T(t, x)$; the corresponding zero-contour $\{x : T(t, x) = 0\}$ is highlighted in red. The other contours show the kernel density estimate of the distribution of the respective grids $\{x_t^n\}$.

Performance Comparison

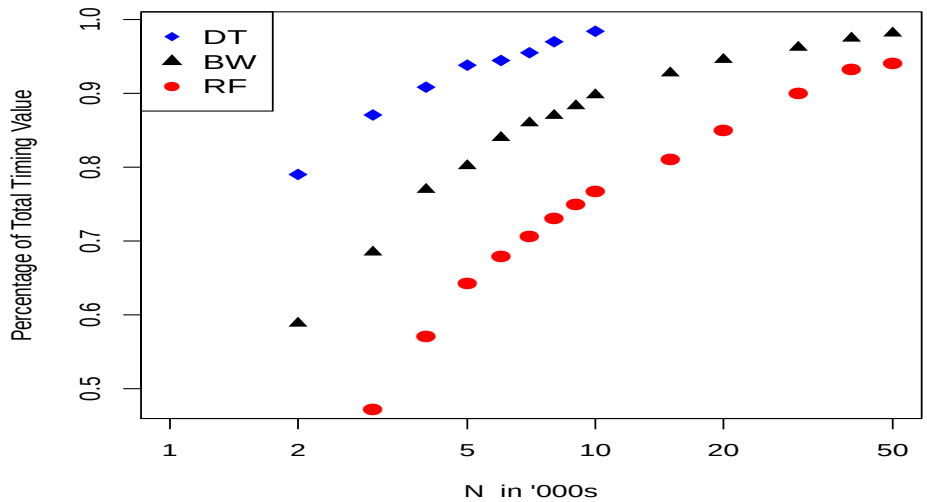


Figure : Performance of (i) a dynamic tree sequential RMC (DT), (ii) localized linear regression (BW) LSMC [1] and (iii) random forest LSMC (RF) methods as a function of grid size N. We plot the percentage of estimated extrinsic value, $\frac{\hat{V}(0,x_0)-\underline{v}(0,x_0)}{V(0,x_0)-\underline{v}(0,x_0)}$, relative to the benchmark $V(0,x_0) = 1.464$ and the intrinsic value $\underline{v}(0,x_0) = \mathbb{E}_{0,x_0}[h_T(X_T)] = 1.230$.

Disclaimer: Total running time of DT is much slower ...

Switching Problems

- (X_t): Markov state process, $t = 0, 1, 2, \ldots$
- $h(x, u)$: reward function
- Action space $u_t \in \mathcal{A} = \{u_1, \ldots, u_i, \}$
- Find greatest expected reward: value function $V(0, x) = \mathbb{E}_{0,x}\left[\sum_{t=0}^{T} h(X_t, u^*(t, X_t))\right]$
- Closed loop feedback strategies $u_t = u^*(t, x)$
- Initial condition is $X_t = x, u_t = u_i$
- Action regions $\mathfrak{S}_i(t) = \{x : u^*(t, x) = u_i\}$

Global Ranking Problem

- As before, using backward induction on t can build up the policy maps to solve the dynamic control problem
- RMC aims to exploit the small action space to focus on policy search vs. usual value function approximation
- The sequential design problem becomes: Given generating models (x, y^k), $k = 1, \ldots, K$, $y^k = \hat{y}^k(x) + \varepsilon^k$ and state space $\mathcal{X}$, how to efficiently estimate the global classifier $\arg\max_k \hat{y}^k(x)$
- At any given site x, only care about the best two alternatives...

Figure : Adaptive comparison of expected costs of Vaccination and Do-Nothing in a stochastic SIR model. Vaccination reduces infection rate but has a high fixed cost.

Take Aways & Current Research

1. RMC is really about contour-finding
2. There is tremendous potential in optimizing the stochastic grids used in RMC

- Both tasks can be efficiently done using recently developed sequential modeling tools
- Opens the door to many related computational statistics improvements
- Novel sequential design setting: high throughput, low signal-to-noise, contour-finding problem

Q: When is the boundary complexity $\ll$ value function complexity?

Take Aways & Current Research

1. RMC is really about contour-finding
2. There is tremendous potential in optimizing the stochastic grids used in RMC

- Both tasks can be efficiently done using recently developed sequential modeling tools
- Opens the door to many related computational statistics improvements
- Novel sequential design setting: high throughput, low signal-to-noise, contour-finding problem

Q: When is the boundary complexity $\ll$ value function complexity?

THANK YOU!

References

B. Bouchard and X. Warin, *Monte-Carlo valorisation of American options: facts and new algorithms to improve existing methods*, in Numerical Methods in Finance, R. Carmona, P. D. Moral, P. Hu, and N. Oudjane, eds., vol. 12 of Springer Proceedings in Mathematics, Springer, 2011.

R. Gramacy, M. Taddy, and N. Polson, *Dynamic trees for learning and design*, Journal of the American Statistical Association, (2011).
to appear; preprint on arXiv:0912.1636.

M. Kohler, *A review on regression-based Monte Carlo methods for pricing American options*, in Recent Developments in Applied Probability and Statistics, Springer, 2010, pp. 37–58.

Bect, J., Ginsbourger, D., Li, L., Picheny, V., and Vazquez, E., *Sequential design of computer experiments for the estimation of a probability of failure*, Statistics and Computing, 22(3), 773–793, (2012).

R. Gramacy and M. Ludkovski *Sequential Regression for Optimal Stopping Problems* preprint, 2013.