Efficient Importance Sampling

David N. DeJong
University of Pittsburgh

Spring 2008

Overview

Our goal is to calculate integrals of the form

$$G(Y) = \int_{\Theta} \varphi(\theta; Y) \, d\theta.$$

Special case (e.g., posterior moment):

$$G(Y) = \int_{\Theta} \phi(\theta; Y) \, p(\theta | Y) \, d\theta.$$

Scenario: analytical solutions to these integrals are unavailable. We will remedy this problem using **numerical approximation methods**.

Overview, cont.

In the context of achieving likelihood evaluation and filtering in state space representations, recall that we face the challenge of constructing the filtering density

$$f\left(s_t | Y_t\right) = \frac{f\left(y_t, s_t | Y_{t-1}\right)}{f\left(y_t | Y_{t-1}\right)} = \frac{f\left(y_t | s_t, Y_{t-1}\right) f\left(s_t | Y_{t-1}\right)}{f\left(y_t | Y_{t-1}\right)},$$

where

$$f\left(s_t | Y_{t-1}\right) = \int f\left(s_t | s_{t-1}, Y_{t-1}\right) f\left(s_{t-1} | Y_{t-1}\right) ds_{t-1},$$

and

$$f\left(y_t | Y_{t-1}\right) = \int f\left(y_t | s_t, Y_{t-1}\right) f\left(s_t | Y_{t-1}\right) ds_t.$$

Overview, cont.

To gain intuition, suppose the integral we face is of the form

$$G(Y) = \int_{\Theta} \phi\left(\theta; Y\right) p\left(\theta | Y\right) d\theta,$$

and it is possible to obtain pseudo-random drawings θ_i from $p\left(\theta | Y\right)$. Then by the law of large numbers,

$$\overline{G(Y)}_N = \frac{1}{N}\sum_{i=1}^{N}\phi\left(\theta_i; Y\right)$$

converges in probability to $G(Y)$. We refer to $\overline{G(Y)}_N$ as the **Monte Carlo** estimate of $G(Y)$. As we shall see, from the standpoint of **numerical efficiency**, this represents a **best-case scenario**.

Overview, cont.

The MC estimate of the standard deviation of $G(Y)$ is given by

$$\overline{\sigma}_N\left(G(Y)\right) = \left[\left(\frac{1}{N}\sum_i^N \phi\left(\theta_i; Y\right)^2\right) - \overline{G\left(Y\right)}_N^2\right]^{1/2}.$$

The numerical standard error associated with $\overline{G(Y)}_N$ is given by

$$s.e.\left(\overline{G\left(Y\right)}_N\right) = \frac{\overline{\sigma}_N\left(G(Y)\right)}{\sqrt{N}}.$$

Thus for $N = 10,000$, $s.e.\left(\overline{G\left(Y\right)}_N\right)$ is 1% of the size of $\overline{\sigma}_N\left(G(Y)\right)$.

Importance Sampling (Geweke, 1989 Econometrica)

If $p(\theta|Y)$ is unavialable as a sampler, one remedy is to augment the targeted integrand with an **importance sampling** distribution $g(\theta|a)$:

$$\begin{aligned} G(Y) &= \int_{\Theta} \frac{\varphi(\theta; Y)}{g(\theta|a)} g(\theta|a)\, d\theta \\ &= \int_{\Theta} \frac{\phi(\theta; Y)\, p(\theta|Y)}{g(\theta|a)} g(\theta|a)\, d\theta. \end{aligned}$$

Key requirements:

- Support of $g(\theta|a)$ must span that of $\varphi(\theta|Y)$
- $E[G(Y)]$ must exist and be finite.
- $g(\theta|a)$ must be implementable as a sampler

Importance Sampling, cont.

- MC estimate of $G(Y)$:

$$\overline{G(Y)}_N = \frac{1}{N}\sum_{i=1}^{N}\omega_i \ ; \qquad \omega_i = \frac{\varphi(\theta_i|Y)}{g(\theta_i|a)}$$

- MC estimate of the standard deviation of ω w.r.t. $g(\theta|a)$:

$$\overline{\sigma}_N(\omega(\theta, Y)) = \left[\left(\frac{1}{N}\sum\omega_i^2\right) - \overline{G(Y)}_N^2\right]^{1/2}.$$

- Numerical standard error associated with $\overline{G(\theta, Y)}_N$:

$$s.e.\left(\overline{G(Y)}_N\right)_I = \frac{\overline{\sigma}_N(\omega(\theta, Y))}{\sqrt{N}}.$$

- **Note:** Variability in ω translates into increased n.s.e. (numerical inefficiency)

Importance Sampling, cont.

For the special case in which the integrand factorizes as

$$\varphi(\theta_i|Y) = \phi(\theta; Y)\, p(\theta|Y),$$

$$\begin{aligned}\overline{G(Y)}_N &= \frac{1}{N}\sum_{i=1}^{N}\phi(\theta_i; Y)\, w_i, \qquad w_i = \frac{p(\theta_i|Y)}{g(\theta_i|a)} \\ \overline{\sigma}_N(G(Y)) &= \left[\left(\frac{1}{N}\sum_{i}^{N}\phi(\theta_i; Y)^2\, w_i\right) - \overline{G(Y)}_N^2\right]^{1/2}.\end{aligned}$$

Importance Sampling, cont.

Continuing with the special case, if we lack an integrating constant for either $p(\theta|Y)$ or $g(\theta|a)$, we can work instead with

$$\overline{G(Y)}_N = \left(\frac{1}{\sum w_i}\right)\sum_{i=1}^{N}\phi(\theta_i; Y)\, w_i$$

$$\overline{\sigma}_N(G(Y)) = \left[\left(\left(\frac{1}{\sum w_i}\right)\sum_{i=1}^{N}\phi(\theta_i; Y)^2\, w_i\right) - \overline{G(Y)}_N^2\right]^{1/2}.$$

The impact of ignoring integrating constants is eliminated by the inclusion of the accumulated weights in the denominators of these expressions.

Importance Sampling, cont.

Regardless of whether $p(\theta|Y)$ and $g(\theta|a)$ are proper p.d.f.s, notice that when $p(\theta|Y)$ itself can be used as a sampling density (i.e., $p(\theta|Y) = g(\theta|a)$), then $w_i = 1 \; \forall \; i$, and we revert to the best-case scenario.

In general, the accuracy of our approximation will fall short of the best-case scenario. To judge the degree of the shortfall, various metrics are available.

Importance Sampling, cont.

Two metrics for judging numerical accuracy:

- $\frac{w_{\max}^2}{\sum w_i^2}$ (good practical benchmark: 1%)
- relative numerical efficiency (**RNE**)

Importance Sampling, cont.

Motivation for RNE. Issue: how close are we to the best-case scenario?

- Under the best-case scenario, recall that n.s.e. is given by

$$s.e.\left(\overline{G\left(Y\right)}_N\right) = \frac{\overline{\sigma}_N\left(G(Y)\right)}{\sqrt{N}}.$$

- Actual n.s.e. is given by

$$s.e.\left(\overline{G\left(Y\right)}_N\right)_I = \frac{\overline{\sigma}_N\left(\omega\left(\theta, Y\right)\right)}{\sqrt{N}},$$

where recall

$$\overline{\sigma}_N\left(\omega\left(\theta, Y\right)\right) = \left[\left(\frac{1}{N}\sum\omega_i^2\right) - \overline{G\left(Y\right)}_N^2\right]^{1/2}, \qquad \omega_i = \frac{\varphi\left(\theta_i|Y\right)}{g\left(\theta_i|a\right)}$$

Importance Sampling, cont.

The idea behind RNE is to compare the actual n.s.e. to an estimate of the optimal (best-case) n.s.e.:

$$\begin{aligned} RNE &= \frac{(\text{ideal n.s.e.})^2}{(\text{actual n.s.e.})^2} \\ &= \frac{\left(\frac{\overline{\sigma}_N(G(Y))}{\sqrt{N}}\right)^2}{\left(s.e.\left(\overline{G(Y)}_N\right)_I\right)^2}. \end{aligned}$$

Importance Sampling, cont.

Rearranging yields

$$s.e.\left(\overline{G\left(Y\right)}_N\right)_I = \frac{\overline{\sigma}_N\left(G(Y)\right)}{\sqrt{N \cdot RNE}}.$$

Note: relative to the best-case scenario, $\sqrt{N \cdot RNE}$ replaces $\sqrt{N}$ in the denominator. Thus the further is RNE from 1, the more draws are required to achieve a given level of accuracy relative to the best-case scenario.

Example

Suppose $p(\theta|Y) \sim N(\mu, \Sigma)$,

$$\mu = \begin{bmatrix} 1 \\ 2 \\ 3 \\ 4 \\ 5 \end{bmatrix}, \qquad corr(\Sigma) = \begin{bmatrix} 1 & 0.6 & 0 & 0 & 0 \\ 0.6 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & -0.8 \\ 0 & 0 & 0 & -0.8 & 1 \end{bmatrix},$$

$$sqrt(diag(\Sigma))' = \begin{bmatrix} 2 & 0.2 & 5 & 1 & 0.1 \end{bmatrix}$$

Statistics of interest:

$$E(sumc(\mu)), \qquad E(prodc(\mu))$$

Example, cont.

Using `rndseed 123456789`, and $N = 10,000$, MC estimates $(\widehat{\mu},\ sqrt\widehat{(diag}(\Sigma)))$, n.s.e.$(\widehat{\mu})$:

1.0077	1.9664	0.0197
2.0031	0.2007	0.0020
2.9157	4.9899	0.0499
4.0041	0.9793	0.0098
4.9993	0.0986	0.0010

Example, cont.

MC estimate of $corr(\Sigma)$:

1.0000	0.6060	0.0044	-0.0139	0.0107
0.6060	1.0000	0.0049	-0.0048	0.0016
0.0044	0.0049	1.0000	-0.0013	-0.0110
-0.0139	-0.0048	-0.0013	1.0000	-0.7969
0.0107	0.0016	-0.0110	-0.7969	1.0000

Example, cont.

MC estimates of statistics (mean, std. dev., n.s.e.(mean)):

$E\left(sumc(\mu)\right)$	14.9299	5.4883	0.0549
$E(prodc(\mu))$	130.7018	533.3792	5.3338

Example, cont.

Exercise: replicate

Hint for exercise.
To obtain draws from $N(\mu, \Sigma)$ distribution:

- `swish = chol(sig)';` swish is lower-diagonal Cholesky decomposition of sig (i.e., sig=swish*swish').
- `draw = mu + swish*rndn(n,1);`

Example, cont.

Suppose instead we seek to obtain estimates using an Importance Sampling density

$$g(\theta|a) \sim N(\mu_I, \Sigma_I),$$

with

$$\begin{aligned} \mu_I &= \mu + 1.5 \cdot sqrt(diag(\Sigma)), \\ \Sigma_I &= diagrv(\Sigma, sqrt(diag(\Sigma))) \end{aligned}$$

Example, cont.

Using `rndseed 123456789`, and $N = 10,000$, MC versus IS estimates $(\widehat{\mu}, \textit{sqrt}(\widehat{\textit{diag}(\Sigma)}), \text{n.s.e.}(\widehat{\mu})$:

1.0077	1.9664	0.0197
2.0031	0.2007	0.0020
2.9157	4.9899	0.0499
4.0041	0.9793	0.0098
4.9993	0.0986	0.0010
1.1646	1.5813	0.4140
1.9884	0.1721	0.0572
3.1161	3.5692	0.6526
4.2545	0.8731	0.2691
4.9947	0.0803	0.0173

Example, cont.

MC versus IS estimate of $corr(\Sigma)$:

1.0000	0.6060	0.0044	-0.0139	0.0107
0.6060	1.0000	0.0049	-0.0048	0.0016
0.0044	0.0049	1.0000	-0.0013	-0.0110
-0.0139	-0.0048	-0.0013	1.0000	-0.7969
0.0107	0.0016	-0.0110	-0.7969	1.0000

1.0000	0.6712	0.0623	-0.2506	-0.2307
0.6712	1.0000	0.2031	-0.2387	0.0065
0.0623	0.2031	1.0000	-0.1987	0.2975
-0.2506	-0.2387	-0.1987	1.0000	-0.6440
-0.2307	0.0065	0.2975	-0.6440	1.0000

Example, cont.

MC versus estimates of statistics (mean, std. dev., n.s.e.(mean)):

$E(sumc(\mu))$	14.9299	5.4883	0.0549
$E(prodc(\mu))$	130.7018	533.3792	5.3338

$E(sumc(\mu))$	15.5184	3.9212	0.8116
$E(prodc(\mu))$	168.4666	471.0707	64.8759

Example, cont.

Accuracy Diagnostics

Summary statistics on weights:

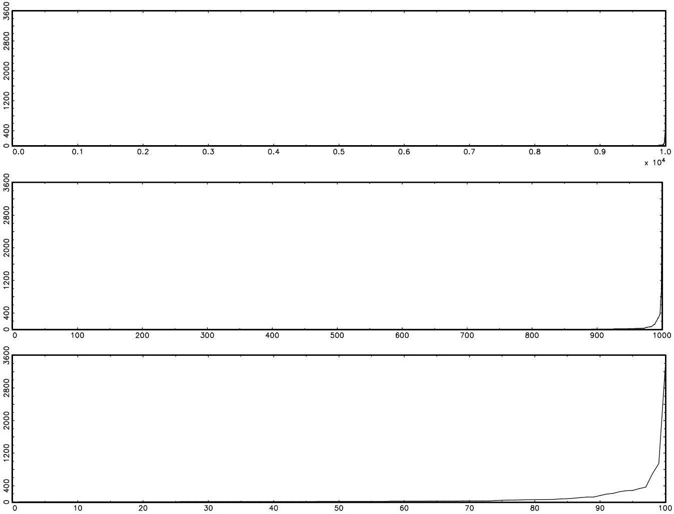
avg,	stdev,	min,	max,	maxsq/totsq:
1.0120	36.3643	0.0000	3341.8561	0.8440

RNEs and 1/RNEs:

0.0023	428.4143
0.0053	189.6681

Example, cont.

Plot of weights (all, top 1,000, top 100):



Example, cont.

Exercise: Replicate

Improving Efficiency

From a programming perspective, two simple approaches to improving efficiency are:

- Increase N (RNEs indicate good rules of thumb for necessary increases). This brute-force method is often computationally prohibitive.
- **Sequential updating.** (Can still be expensive, but less brutish.)

Improving Efficiency, cont.

Sequential updating:

- Begin with an initial parameterization a_0 for $g(\theta|a)$ (e.g., (μ_0, Σ_0)).
- Calculate $\widehat{\theta}_0$, map into a_1.
- Repeat until a_i yields an acceptable level of numerical accuracy.

Improving Efficiency, cont.

Returning to the example, RNEs and 1/RNEs evolve as follows:

Iteration 0:

0.0023	428.4143
0.0053	189.6681

Iteration 1:

0.0575	17.4021
0.0331	30.1702

Iteration 2:

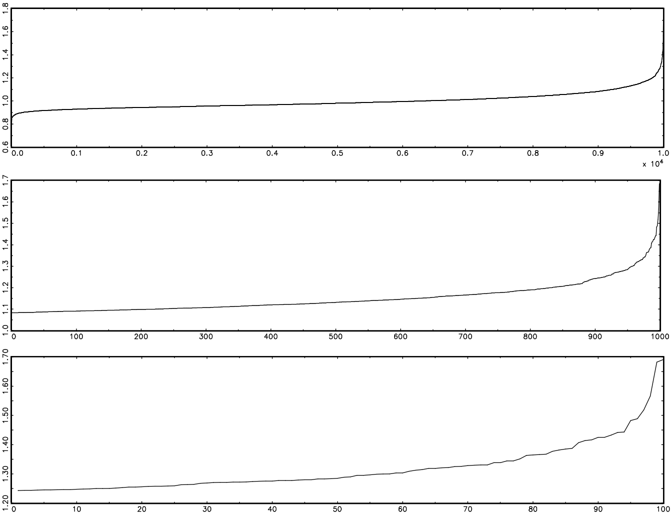
0.8018	1.2472
0.7302	1.3694

Iteration 3:

0.9790	1.0215
0.9535	1.0488

Improving Efficiency, cont.

Weight plots, Iteration 3:



Improving Efficiency, cont.

Caveats regarding sequential updating:

- Convergence to target is not guaranteed
- Performance can be sensitive to starting values
- Initial sampler should be sufficiently diffuse to ensure coverage of appropriate range for targeted integrand

Improving Efficiency, cont.

Aside regarding coverage:
The multivariate-t density is an attractive sampler relative to the normal density: it has similar location and shape parameters, but has **thicker tails**. Tail thickness controlled by degrees-of-freedom parameter v (smaller v, thicker tails).

- Parameters of the multivariat-t: (γ, V, v).
- Mean and second-order moments: γ, $\left(\frac{v}{v-2}\right) V^{-1}$

Improving Efficiency, cont.

Algorithm for obtaining drawings μ_i from multivariate-t (γ, V, v) :

- Obtain s_i from a $\chi^2(v)$ distribution:

$$s_i = \sum_{j=1}^{v} x_j^2, \qquad x_j \sim N(0,1)$$

- Use s_i to construct the scaling factor

$$\sigma_i = (s_i / v)^{-1/2}$$

- Obtain μ_i as

$$\begin{aligned} \mu_i &= \gamma \pm \sigma_i V^{-1/2} w_i, \qquad w_{i,j} \sim N(0,1), \qquad j = 1, ..., k \\ V^{-1/2} &= chol(V^{-1})' \end{aligned}$$

- Note: use of $\pm$ yields *antithetic acceleration* (Geweke, 1988, *JoE*)

Improving Efficiency, cont.

Exercise: Replace the normal densities used as Importance Samplers in the exercise above with multi-t densities, with $\gamma = \mu$ and $V^{-1} = \Sigma$. Experiment with alternative $\upsilon' s$ to assess the impact on numerical efficiency.

EIS (Richard and Zhang, 2007 JoE)

Goal: Tailor $g(\theta|a)$ (via the specification of a) to minimize the n.s.e. associated with the approximation of

$$G(Y) = \int_{\Theta} \varphi(\theta|Y)\, d\theta.$$

Write $g(\theta|a)$ as

$$\begin{aligned} g(\theta|a) &= \frac{k(\theta;a)}{\chi(a)}, \\ \chi(a) &= \int_{\Theta} k(\theta;a)\, d\theta. \end{aligned}$$

EIS, cont.

Details regarding the tailoring of $g(\theta|a)$ are distinct for two special cases:

- $g(\theta|a)$ is parametric (i.e., a normal distribution)
- $g(\theta|a)$ is piecewise-linear

EIS, cont.

When $g(\theta|a)$ is fully parametric, n.s.e. is (approximately) minimized via iterations on

$$\begin{aligned}
(\widehat{a}_{l+1}, \widehat{c}_{l+1}) &= \arg\min_{a,c} \overline{Q}_N(a, c; Y|\widehat{a}_l), \\
\overline{Q}_N(a, c; Y|\widehat{a}_l) &= \frac{1}{N}\sum_{i=1}^{N} d^2\left(\theta_i^l, a, c, Y\right)\omega\left(\theta_i^l; Y, \widehat{a}_l\right), \\
d\left(\theta_i^l, a, c, Y\right) &= \ln\varphi\left(\theta_i^l|Y\right) - c - \ln k\left(\theta_i^l; a\right), \\
\omega\left(\theta_i^l; Y, \widehat{a}_l\right) &= \frac{\varphi(\theta_i|Y)}{g(\theta_i|\widehat{a}_l)}.
\end{aligned}$$

The term c is a normalizing constant that controls for factors in φ and g that do not depend upon θ. Typically, it suffices to set $\omega\left(\theta_i^l; Y, \widehat{a}_l\right) = 1 \ \forall\ i$.

EIS, cont.

When $g(\theta|a)$ is piecewise-linear, the parameters a are grid points:

$$a' = (a_0, ..., a_R), \qquad a_0 < a_1 < ... < a_R.$$

In this case, the kernel $k(\theta; a)$ is given by

$$\begin{aligned} \ln k_j(\theta; a) &= \alpha_j + \beta_j \theta \qquad \forall \theta \in [a_{j-1}, a_j], \\ \beta_j &= \frac{\ln \varphi(a_j) - \ln \varphi(a_{j-1})}{a_j - a_{j-1}}, \quad \alpha_j = \ln \varphi(a_j) - \beta_j a_j. \end{aligned}$$

Optimization is achieved by selecting $\widehat{a}$ as an equal-probability division of the support of $\varphi(\theta|Y)$.

EIS, cont.

Given final estimates $(\widehat{a}, \widehat{c})$, the EIS estimate of $G(Y)$ is given by

$$\overline{G\left(Y\right)}_N = \frac{1}{N}\sum_{i=1}^{N}\omega\left(\theta_i; Y, \widehat{a}\right).$$

N.S.E. is computed as indicated above.

Implementation, Gaussian Sampler

To simplify notation, denote the targeted integrand as $\varphi(\theta|Y) \equiv \varphi(\theta)$. We'll take θ as $k-$dimensional, with elements $(x_1, x_2, ...x_k)$.

With $g(\theta|a)$ Gaussian, a consists of the $k \times 1$ vector of means μ and the $k \times k$ covariance matrix Σ. Since the covariance matrix is symmetric, the number of auxiliary parameters reduces to $k + k(k+1)/2$.

The precision matrix $H = \Sigma^{-1}$.

Implementation, Gaussian Sampler, cont.

Our goal is to choose (μ, H) to approximate optimally $\ln \varphi(s)$ by a Gaussian kernel:

$$\begin{aligned} \ln \varphi(\theta) &\propto -\frac{1}{2}(\theta-\mu)' H(\theta-\mu) \\ &\propto -\frac{1}{2}\left(\theta' H \theta - 2\theta' H \mu\right). \end{aligned}$$

Recall that by 'optimally', we refer to the weighted-squared-error minimization introduced above.

Implementation, Gaussian Sampler, cont.

The term $\theta' H \theta$ can be written as

$$\begin{pmatrix} x_1 & x_2 & . & . & x_k \end{pmatrix} \begin{pmatrix} h_{11} & h_{21} & . & . & h_{1k} \\ h_{21} & h_{22} & . & . & h_{2k} \\ . & . & . & . & . \\ . & . & . & . & . \\ h_{k1} & h_{k2} & . & . & h_{kk} \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ . \\ . \\ x_k \end{pmatrix}.$$

Implementation, Gaussian Sampler, cont.

Expanding $\theta' H\theta$, we obtain

$$\begin{aligned}\theta' H\theta &= h_{11}\left(x_1^2\right) + h_{22}\left(x_2^2\right) + ... + h_{jj}\left(x_{kj}^2\right) \\ &+2h_{21}\left(x_2x_1\right) + 2h_{31}\left(x_3x_1\right) + ... + 2h_{k1}\left(x_kx_1\right) \\ &+2h_{32}\left(x_3x_2\right) + 2h_{42}\left(x_4x_2\right) + ... + 2h_{k2}\left(x_kx_2\right) \\ &... \\ &+2h_{k(k-1)}\left(x_kx_{k-1}\right).\end{aligned}$$

This expression indicates that the coefficients of the squares, pairwise products and the individual components of θ are in one-to-one correspondence with the μ and H.

Implementation, Gaussian Sampler, cont.

Given this correspondence, the EIS optimization problem amounts to a weighted-least-squares problem involving the regression of $\ln \varphi(s)$ on

$$\left[1,\ x_1, ..., x_k,\ x_1 x_2, x_1 x_3, ..., x_{k-1} x_k\ x_1^2, x_2^2, ..., x_k^2\right].$$

The number of regressors is $K = \left(1 + k + \frac{k(k+1)}{2}\right)$.

Implementation, Gaussian Sampler, cont.

Implementation Algorithm

- Specify a_0, generate

$$y = \begin{pmatrix} \ln \varphi(\theta_1) \\ \dots \\ \ln \varphi(\theta_M) \end{pmatrix}, \qquad w = \begin{pmatrix} \omega_1 \\ \dots \\ \omega_M \end{pmatrix}, \qquad \frac{\varphi\left(\theta_i | Y\right)}{g\left(\theta_i | a_0\right)},$$

$$\begin{aligned} X &= \begin{pmatrix} \kappa_1 \\ \dots \\ \kappa_M \end{pmatrix}, \\ \kappa_i &= \left[1 \tilde{\ } \theta_i' \tilde{\ } vech(\theta_i \cdot \theta_i')'\right]. \end{aligned}$$

(Note: $M << N$)

Implementation, Gaussian Sampler, cont.

Implementation Algorithm, cont.

- Construct

$$\widetilde{y} = y. * (w.\hat{}2), \qquad \widetilde{X} = X. * (w.\hat{}2).$$

(Caution: set $w = 1$ when using a poor initial sampling density.)

- Estimate

$$\widehat{\beta} = (\widetilde{X}'\widetilde{X})^{-1}\left(\widetilde{X}\widetilde{y}\right).$$

- Map $\widehat{\beta}$ into $\left(\widehat{\mu}, \widehat{\Sigma}\right)$. Jointly, these constitute a_1.
- Replacing a_0 above with a_1, repeat until convergence.

Implementation, Gaussian Sampler, cont.

Implementation Algorithm, cont.

To map $\widehat{\beta}$ into $\left(\widehat{\mu}, \widehat{\Sigma}\right)$:

- Map the $k+2$ through K elements of $\widehat{\beta}$ into a symmetric matrix $\widetilde{H}$, with j^{th} diagonal element corresponding to the coefficient associated with the squared value of the j^{th} element of θ, and $(j,k)^{th}$ element corresponding to the product of the j and k^{th} element of θ. I.E.,

$$\widetilde{H}\texttt{=xpnd(}\widehat{\beta}\texttt{[k+2:rows(beta)]);}$$

- Construct $\widetilde{\widetilde{H}}$ by multiplying all elements of $\widetilde{H}$ by -1, then multiplying the diagonal elements by 2.
- $\widehat{\Sigma} = \widetilde{\widetilde{H}}^{-1}$
- $\widehat{\mu} = \widehat{\Sigma} \cdot \widehat{\beta}[2 : k+1]$

Implementation, Gaussian Sampler, cont.

Exercise: Return to the example outlined above. Using an initial sampler specified with

$$\begin{aligned} \mu_0 &= \mu + 3 \cdot sqrt(diag(\Sigma)), \\ \Sigma_0 &= 10 \cdot diagrv(\Sigma, sqrt(diag(\Sigma))), \end{aligned}$$

show that the EIS algorithm yields

$$\widehat{\mu} = \mu, \qquad \widehat{\Sigma} = \Sigma$$

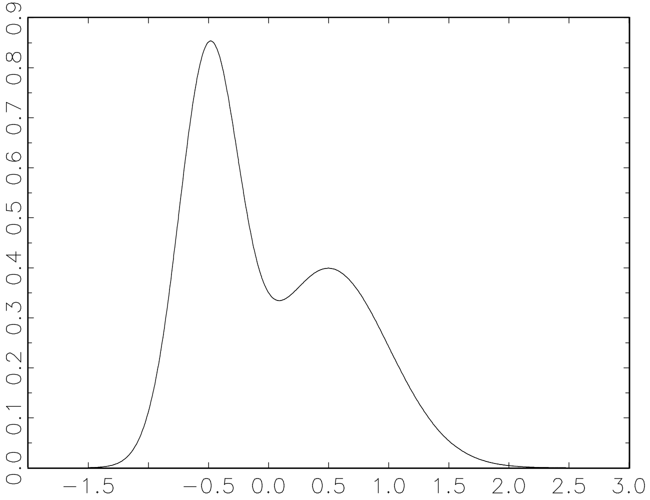
in one step using $M = 50$, $w = 1$. Experiment with alternative initial samplers.

Implementation, Piecewise-Linear Approximation

Context: Effective for use in univariate cases featuring obvious deviations from normality.
(Curse of dimensionality renders implementation problematic in high-dimensional cases.)

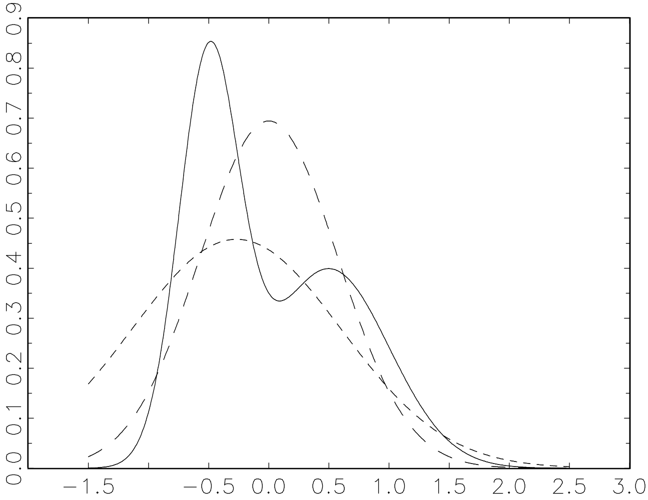
Implementation, PW-L Approx., cont.

Motivation 1. Mixture of Normals, $(\mu, \sigma) = (-0.5, 0.25)$, $(0.5, 0.5)$, equal weights



Implementation, PW-L Approx., cont.

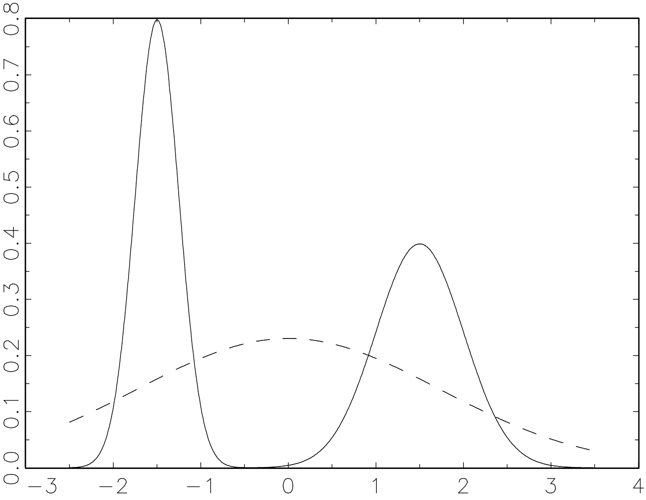
Approximation via Gaussian samplers (handmade and EIS):



RNEs for calculating E(x): 0.61, 0.78.

Implementation, PW-L Approx., cont.

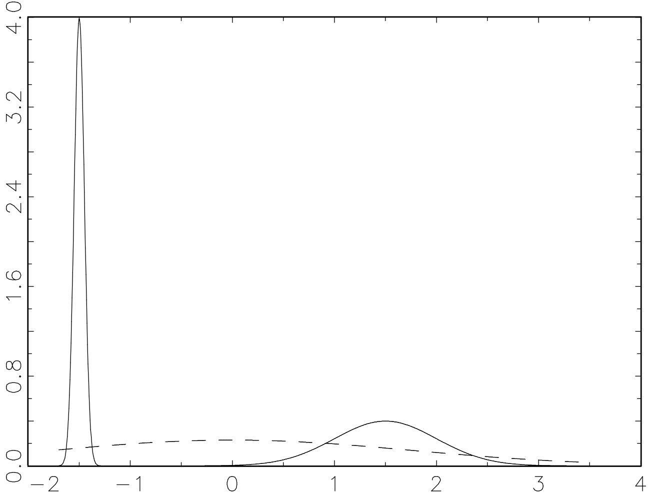
Motivation 2. $(\mu, \sigma) = (-1.5, 0.25)$, $(1.5, 0.5)$, equal weights



RNE: 0.37

Implementation, PW-L Approx., cont.

Motivation 3. $(\mu, \sigma) = (-1.5, 0.05)$, $(1.5, 0.5)$, equal weights



RNE: 0.11

Implementation, PW-L Approx., cont.

Reboot: Recall that when $g\left(\theta|a\right)$ is piecewise-linear, the parameters a are grid points:

$$a' = (a_0, ..., a_R), \qquad a_0 < a_1 < ... < a_R.$$

In this case, the kernel $k(\theta; a)$ is given by

$$\begin{aligned} \ln k_j(\theta; a) &= \alpha_j + \beta_j \theta \qquad \forall \theta \in [a_{j-1}, a_j], \\ \beta_j &= \frac{\ln \varphi(a_j) - \ln \varphi(a_{j-1})}{a_j - a_{j-1}}, \quad \alpha_j = \ln \varphi(a_j) - \beta_j a_j. \end{aligned}$$

Optimization is achieved by selecting $\widehat{a}$ as an equal-probability division of the support of $\varphi\left(\theta|Y\right)$.

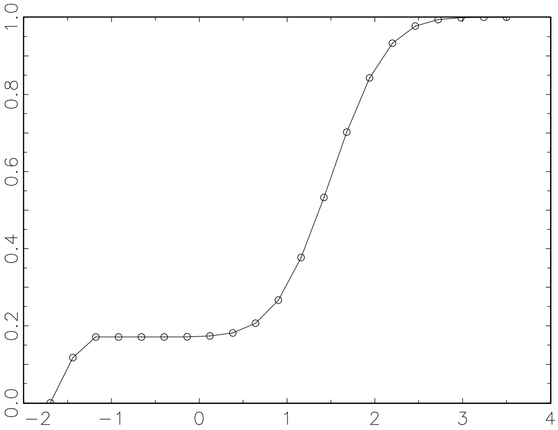
Implementation, PW-L Approx., cont.

In order to achieve equal-probability division, and to implement the distribution as a sampler, we must

- construct its associated CDF
- invert the CDF.

Implementation, PW-L Approx., cont.

To gain intuition behind implementation and inversion, consider the CDF associated with Case 3:



Inversion involves inducing a mapping from the y to the x axis. Implementation involves mapping drawings obtained from a $U[0,1]$ distribution onto the x axis.

Implementation, PW-L Approx., cont.

Letting s denote the $x-$axis variable, the **CDF** of k can be written as

$$\begin{aligned} K_j(s;a) &= \frac{\chi_j(s;a)}{\chi_n(a)}, \quad \forall s \in [a_{j-1}, a_j], \\ \chi_j(s;a) &= \chi_{j-1}(a) + \frac{1}{\beta_j}\left[k_j(s;a) - k_j(a_{j-1};a)\right], \\ \chi_0(a) &= 0, \quad \chi_j(a) = \chi_j(a_j;a). \end{aligned}$$

Note: $\chi_n(a)$ is the integrating constant associated with the pdf.

Implementation, PW-L Approx., cont.

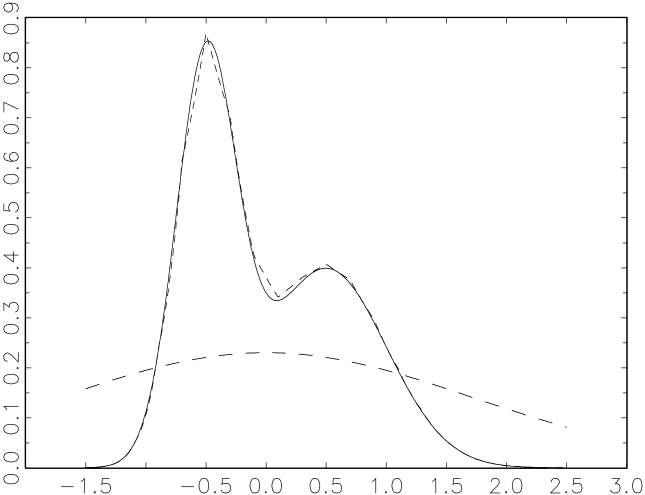
Inversion/implementation:

$$s = \frac{1}{\beta_j}\left\{\ln\left[k_j(a_{j-1};a) + \beta_j\left(u \cdot \chi_R(a) - \chi_{j-1}(a)\right)\right] - \alpha_j\right\},$$

$$u \in \left]0,1\right[\quad \text{and} \quad \chi_{j-1}(a) < u \cdot \chi_R(a) < \chi_j(a).$$

Implementation, PW-L Approx., cont.

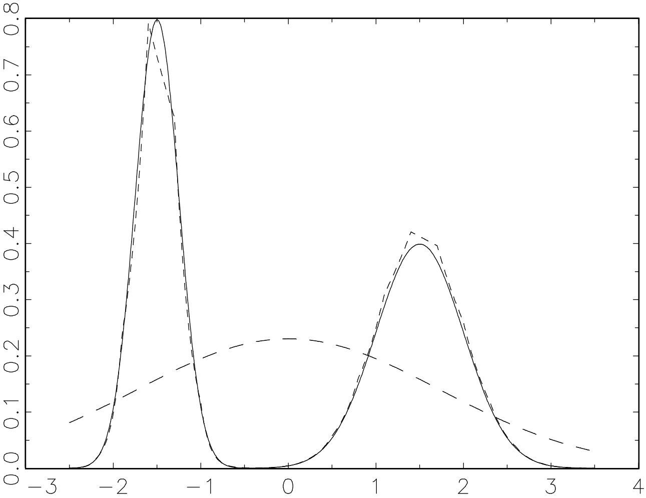
Unrefined (equal spacing over x range) piecewise approximation for case 1:



RNE: 0.99

Implementation, PW-L Approx., cont.

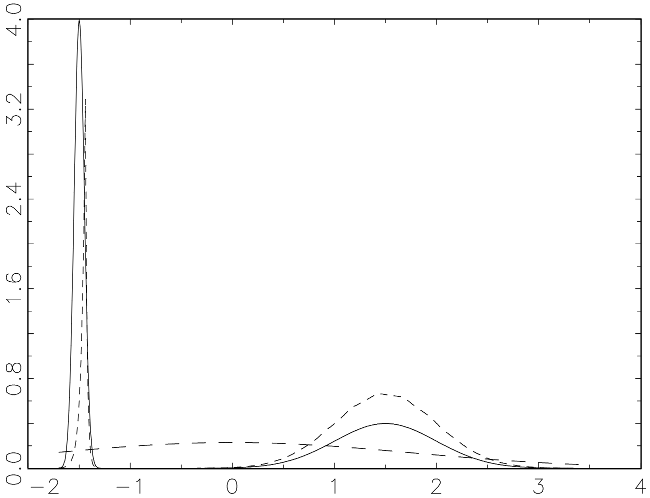
Unrefined (equal spacing over x range) piecewise approximation for case 2:



RNE: 0.99

Implementation, PW-L Approx., cont.

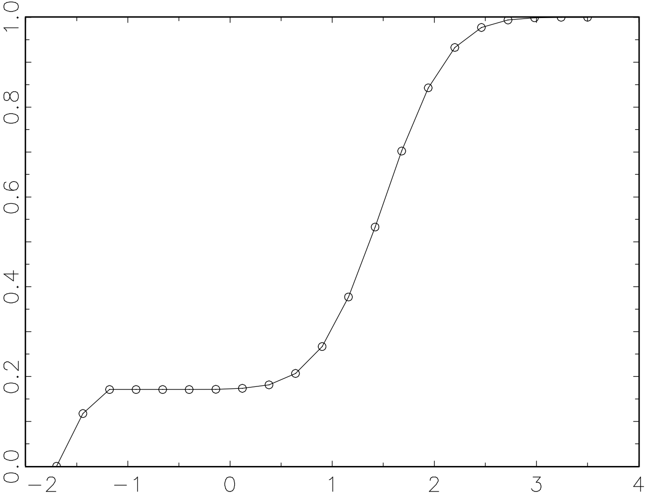
Unrefined (equal spacing over x range) piecewise approximation for case 3:



RNE: 0.24

Implementation, PW-L Approx., cont.

To understand the source of the problem in Case 3, consider again the CDF of the unrefined sampler:



Note: relatively few points along steep portions of the CDF,

Implementation, PW-L Approx., cont.

To address the problem: **equal probability division** of the range for s. I.e., divide the vertical axis of the CDF into equal portions, then map into s :

$$u_i = \varepsilon + (2 - \varepsilon)\frac{i}{R}, \qquad i = 1, ..., R - 1,$$

with ε sufficiently small (typically $\varepsilon = 10^{-4}$) to avoid tail intervals of excessive length.

Implementation, PW-L Approx., cont.

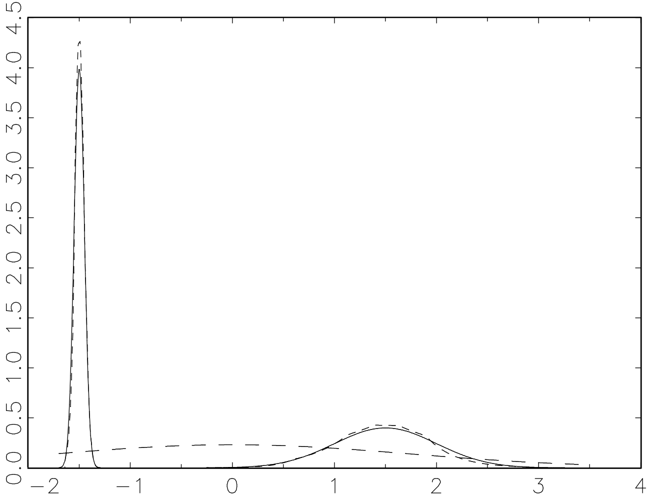
Iterative construction:
Given the step-l grid $\widehat{a}^l$, construct the density kernel k and its CDF K as described above. The step-$l+1$ grid is then computed as

$$\widehat{a}_i^{l+1} = K^{-1}(u_i), \qquad i = 1, ..., R-1.$$

Iterate until (approximate) convergence.

Implementation, PW-L Approx., cont.

Refined approximation for Case 3:



RNE: 0.95

Implementation, PW-L Approx., cont.

Refined CDF: