Bayesian Estimation with Sparse Grids

Kenneth L. Judd and Thomas M. Mertens

Institute on Computational Economics
August 07, 2007

Outline

Introduction

- Bayesian estimation involves integration of the posterior.
- We present a numerical quadrature approach for Bayesian estimation.
- We use sparse grids to deal with high dimensionality.
- It is an alternative to the use of simulations.

Bayesian Estimation

- Get data Y.
- Obtain likelihood function $\mathscr{L}(Y|\theta)$.
- Use prior information $p(\theta)$.
- Posterior: $p(\theta|Y) \sim \mathscr{L}(Y|\theta) \cdot p(\theta)$
- We then compute Bayesian estimators:

$$M = \int h(\theta)\mathscr{L}(Y|\theta)p(\theta)d\theta$$

Introduction

- Main difficulty is dimensionality of the problem.
- Monte-Carlo methods converge at a rate *independent* of dimension — but slowly.
- We want a faster method.
- There is hope: the function is *very* smooth.

Introduction

- Pseudo-random schemes give you convergence $O(N^{-\frac{1}{2}})$.
- Equidistributional sequences give convergence of order 1 for C^1-functions.
- There is convergence of $O(N^{-k})$ for periodic C^k-functions.
- For very smooth functions, there is essentially no "curse of dimensionality".

Related literature

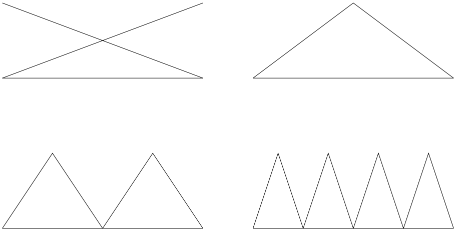
- MCMC: Peter Rossi, "Bayesian Statistics and Marketing"
- Sparse grids: Bungartz and Griebel: "Sparse grids", Acta Numerica (2004).
- Sparse grids have been used in economics and finance.
- Examples in economics are: Kuebler and Krueger, and Winschel.

Approximation

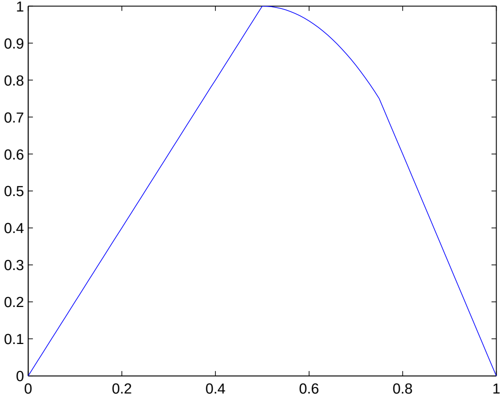
- Goal: Approximate function $f(x)$ with basis functions.
- We can the write the approximation $\hat{f}(x)$ as:

$$\hat{f}(x) = \sum_{(l,i)} u_{(l,i)} \phi_{(l,i)}(x) \approx f(x)$$

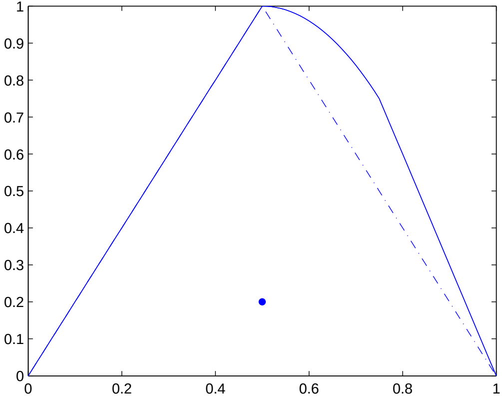
- Basis functions could for instance be:



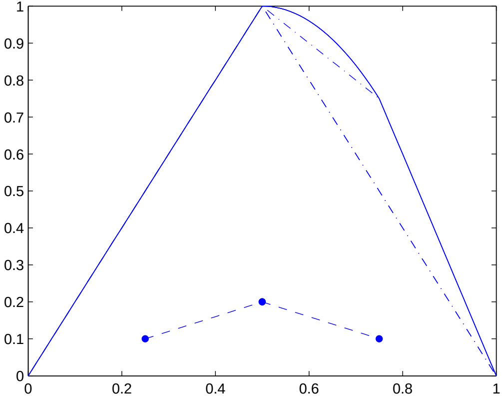
Approximation



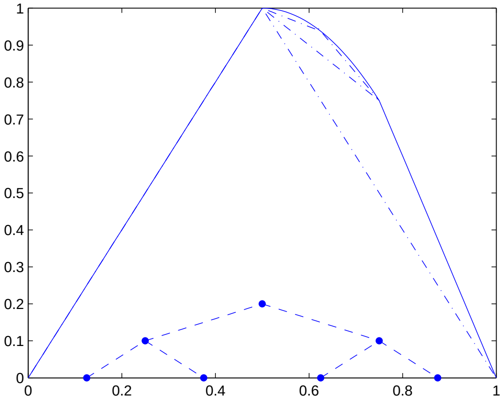
Approximation



Approximation



Approximation



Exponents

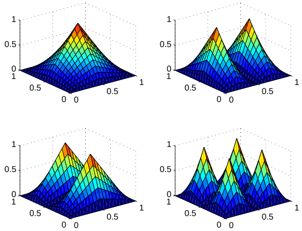
- Why can we truncate the sum at level l?
- Look at the approximation again.
- The absolute value of coefficients shrinks to zero.
- More precisely: $u_{(\mathbf{l},\mathbf{i})} \leq c \cdot 2^{-2\cdot|\mathbf{l}|_1}$.

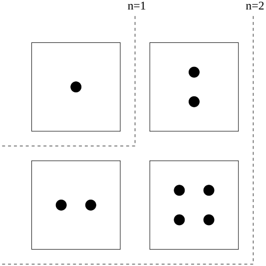
Sparse grids — Construction

- The goal is to generalize the one-dimensional approximation.
- This requires to specify:
  - the basis function
  - the grid

Sparse grids — Construction

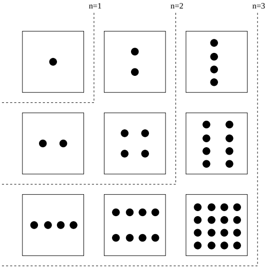
- Choice of basis functions and associated grid.

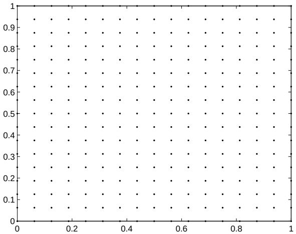




$$\phi_{(\mathbf{l},\mathbf{i})} = \prod_j \phi_{(l_j,i_j)}$$

Sparse grids — Construction

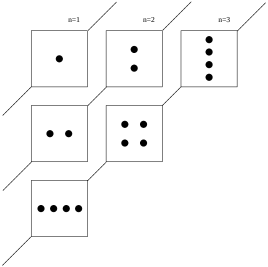


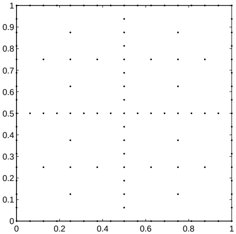


- "Curse of Dimensionality" for full grids — grid size grows at n^d.

Sparse grids — Construction

Sparse grids:





Approximation

- Note: We can now still write the approximation as

$$\hat{f}(\mathbf{x}) = \sum_{(\mathbf{l},\mathbf{i})} u_{(\mathbf{l},\mathbf{i})} \phi_{(\mathbf{l},\mathbf{i})}(\mathbf{x}) \approx f(\mathbf{x})$$

- where $\mathbf{l} = (l_1, \ldots, l_d)$, $\mathbf{i} = (i_1, \ldots, i_d)$
- $\phi_{(\mathbf{l},\mathbf{i})} = \prod_j \phi_{(l_j,i_j)}$
- It is a generalization of the 1-d concept.
- But: sparse grids come at a cost: you have to have bounded second mixed derivatives.

Features

	Full grids	Sparse grids
# grid points	n^d	$O(n \log(n)^{d-1})$
Accuracy		
L_2-error	$O(N^{-2})$	$O(N^{-2} \cdot \log(N)^{d-1})$
L_∞-error	$O(N^{-2})$	$O(N^{-2} \cdot \log(N)^{d-1})$

Why is that such a big thing?

- Just imagine: 32 grid points per dimension, 30 dimensions.
- Sparse grids: $4,518,180$ grid points.
- That fits into a computer's RAM.
- 1427247692705959881058285969449500000000000000 points in full grids don't!

Integration and moments

- Having the approximation, we can now integrate
- Just sum up!

$$\int \sum_{(l,i)} u_{(l,i)} \phi_{(l,i)}(x) = \sum_{(l,i)} u_{(l,i)} \int \phi_{(l,i)}(x)$$

- How about computing moments?
- Use the same grid: since $\|f - \hat{f}\| < \delta$

$$\| \int x^2 (f(x) - \hat{f}(x)) dx \| < \delta \int x^2 dx$$

- Let's look at the problem again.
- Compute expected value of functions of θ

$$M = \int h(\theta)\mathscr{L}(Y|\theta)p(\theta)d\theta$$

- Problem: cannot sample from $\mathscr{L}(Y|\theta)p(\theta)$ directly.
- Use MCMC methods: Gibbs or Metropolis-Hastings.

Technique: MCMC

- Create Markov-chain that has true distribution as stationary distribution.
- First: Gibbs sampling.
- Pick starting point.
- Pick successively draw from each marginal distribution to get to the next step.

Technique: MCMC

- Now: Metropolis-Hastings.
- Pick starting point.
- Draw sample ψ from proposal distribution $Q(\theta'_t|\theta_{t-1})$ (i.e. $Q(\theta'_t|\theta_{t-1}) \sim N(\theta_{t-1}, \Sigma)$).
- Accept draw only if $u < \frac{p(\psi)\mathscr{L}(Y|\psi)}{p(\theta_{t-1})\mathscr{L}(Y|\theta_{t-1})}$ where $u \sim U([0,1])$ otherwise $\theta_t = \theta_{t-1}$.

Technique: MCMC

- Both Gibbs and Metropolis-Hastings samplers have a burn-in phase.
- These are the samples you need to get to the stationary distribution.
- These draws are not used to compute the integral.

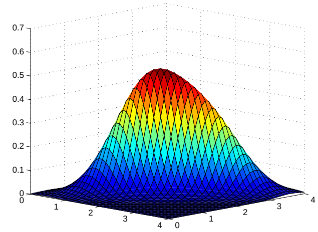
Technique: MCMC

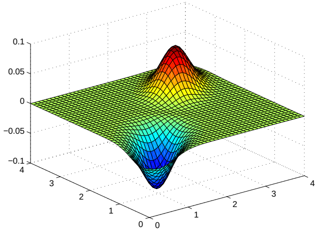
- There are further complications with MCMC
- doesn't use all information about the density at a draw.
- takes long to converge.
- error estimation is comparably hard.
- no possibility to check for identification.

Our approach

- Take "burn-in" samples.
- Find peak of the posterior using samples as starting points.
- Compute Hessian at the peak.
- Take out Gaussian function.
- Repeat until only little mass is left.
- Treat rest with sparse grids.

In pictures

- 

- 

- Treat remainder with sparse grids.

Advantages

- We then have the approximation:

$$p(\theta|Y) = \sum G(\theta) + \sum_{(\mathbf{l},\mathbf{i})} u_{(\mathbf{l},\mathbf{i})}\phi_{(\mathbf{l},\mathbf{i})}(\theta)$$

- We can then integrate since we know

$$\sum \int h(\theta) G(\theta)$$

and

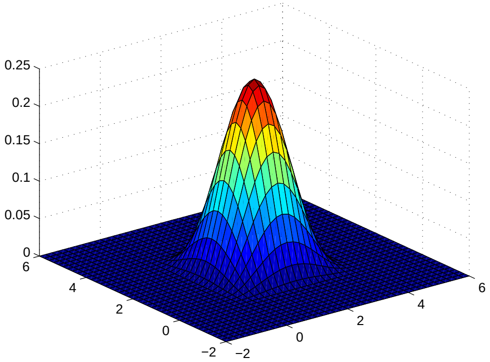
$$\int \sum_{(\mathbf{l},\mathbf{i})} h(\theta) u_{(\mathbf{l},\mathbf{i})}\phi_{(\mathbf{l},\mathbf{i})}(\theta)$$

Advantages

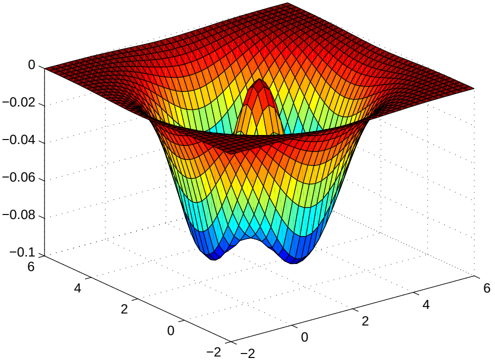
- Precise error estimation for normalized posterior.
- Otherwise have to rely on relative accuracy.
- From one approximation, you can compute all moments.
- You get a normal approximation for free.
- Check for identification.

Numerical results

$$f(\mathbf{x}) = \prod_{i=1}^{d} \frac{1}{2s}\left[1 + cos\left(\frac{x(i)-\mu}{s}\pi\right)\right]$$



Numerical results

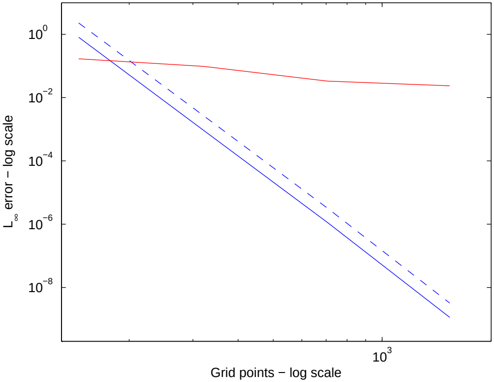


Integration problem

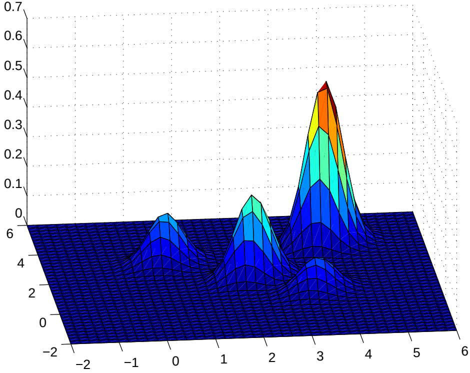
# points	145	321	705	1537
Sparse grids	0.8083	9.26e-4	1.14e-6	1.13e-9
MC	0.1685	0.0974	0.0334	0.0236

Table: L_∞-error quadrature versus MC, 2D

Numerical results



"Sum of normals"

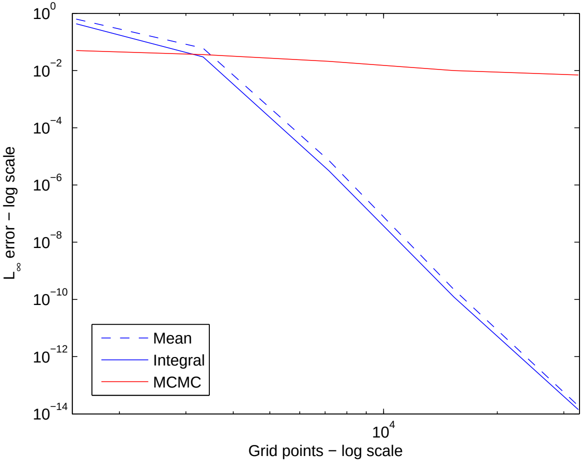


Integration problem

# points	1537	3329	7169	15361	32769
Sparse grids	0.43	0.03	3.12e-006	1.21e-010	1.41e-014
MCMC	0.05	0.036	0.021	0.01	0.007

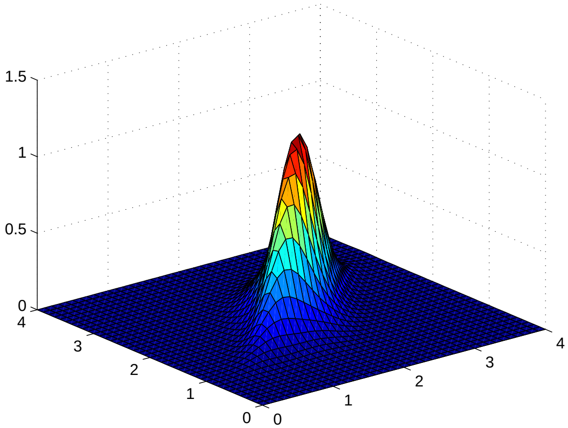
Table: Error for MCMC versus integration, 2D

Numerical results

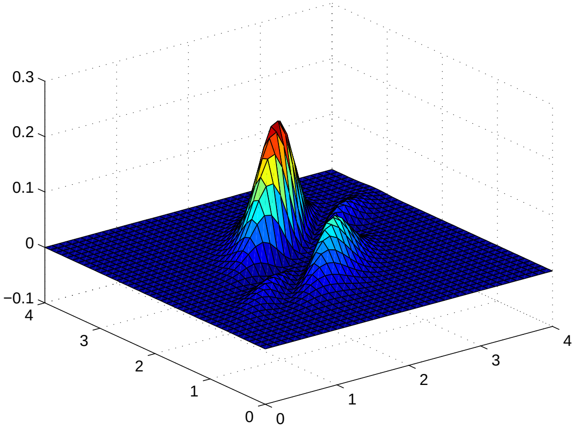


Numerical results

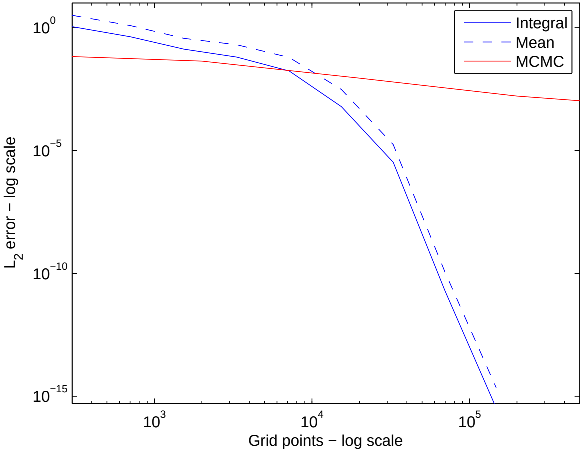
$$f(x) = a \cdot e^{-\frac{1}{2}\cdot(x-\mu)\Sigma(x-\mu)} + b \cdot e^{(x-\mu)^2\Sigma(x-\mu)^2}$$



Numerical results



Numerical results



Result

- This solves three problems:
- 1. Method is fast.
- 2. Error estimation is accurate.
- 3. Get posterior to check for identification.

Conclusion

- Frank Schorfheide said at the macro annual:
- "This is heavy computing: Don't try this at home!"
- We'll put a Matlab code on the web.
- Please do try it at home!
- We can improve along many dimensions: polynomial exactness, better code, and adaptivity.

Applications

- Sparse grids have been used in several contexts.
- Numerical Integration
- Projection methods (e.g. Kuebler and Krueger)
- Solution to partial differential equations