MEETING THE CHALLENGES

OF SOLVING HIGH-DIMENSIONAL MODELS

O' Curse of Dimensionality, Where is Thy Sting?

Kenneth L. Judd

Hoover Institution, University of Chicago, and NBER

September 1, 2007

Curse of Dimensionality

- Many economic models are high dimensional
  - Dynamic Optimization: Multiple kinds of capital stocks
  - DSGE: Multiple consumers/firms/countries
  - Games: Multiple players and states
  - Bayesian analyses compute high-dimensional integrals
- Claim: "You can't solve your model because of the curse of dimensionality."
  - Response I: Analyze silly models
    - Reduce heterogeneity in tastes, abilities, age, etc.
    - Assume no risk
    - Assume common information, common beliefs, etc.
  - Response II: Do bad math
  - Response III: Do bad math while analyzing silly models

- The message today: "The curse is not so bad"
  - I will use our stochasic growth problem as an example, but these comments apply to many other problems.
  - "Theorems" about the curse are irrelevant for economics
  - There are many underutilized tools from math that can help
  - Sensible modelling choices can avoid curse
  - Mathematicians are currently developing tools to tackle the curse
  - Physicists are working to build computers that will avoid the curse
  - If the Boston Red Sox can beat the "Curse of the Bambino" then economists can beat the "Curse of Dimensionality"

Specific Responses to the Challenges of Dimensionality

- Math Tools
  - Compute derivatives efficiently
  - Approximate functions efficiently
  - Choose an efficient domain
  - Approximate integrals efficiently
  - Functional analysis
- Modelling Suggestions
  - Use continuous time
  - Get rid of kinks
  - Specify finite-dimension models
- Look to the future
  - Use "experimental mathematics" - Monte Carlo people do it all the time, why not us?
  - Learn parallel computing tools, and other high power computing architectures
  - Study Griebel-Wozniakowski Theorem
  - Quantum computing

AVAILABLE MATH TOOLS

FOR MULTIDIMENSIONAL PROBLEMS

- Most economists use methods motivated by one-dimensional methods when they solve multidimensional models; the result is inefficient
- There are many tools that are not critical for one-dimensional problems but are powerful for multidimensional problems

Math Tool I: Evaluate Derivatives Efficiently

- Derivatives are important in perturbation methods and in any method that uses nonlinear equation solvers.
  - Analytic derivatives are slow

Analytic Derivatives

		+,-	*,÷	Power	Total flops	Total time
function	$u=(x^\sigma+y^\sigma+z^\sigma)^\rho$	2	0	4	6	22
gradient	$u_x=\sigma\rho x^{\sigma-1}(x^\sigma+y^\sigma+z^\sigma)^{\rho-1}$	4	3	5		32
	$u_y=\sigma\rho y^{\sigma-1}(x^\sigma+y^\sigma+z^\sigma)^{\rho-1}$	4	3	5		32
	$u_z=\sigma\rho z^{\sigma-1}(x^\sigma+y^\sigma+z^\sigma)^{\rho-1}$	4	3	5		32
grad. total:		12	9	15	36	114
Hessian						$\geq$400

– Finite differences are slow

Finite Difference Derivatives

		+,-	*,÷	Power	Total flops	Total time
function	$u = (x^\sigma + y^\sigma + z^\sigma)^\rho$	2	0	4	6	22
gradient	$u_x = (u(x+\Delta, y, z) - u)/\Delta$	3	1	4		24
	$u_y = (u(x, y+\Delta, z) - u)/\Delta$	3	1	4		24
	$u_z = (u(x, y, z+\Delta) - u)/\Delta$	3	1	4		24
grad total:		9	3	12	24	72
Hessian						$\geq$150

- Automatic Differentiation to the rescue!

		+,-	*,÷	a^b	Total flops	Appx. clock time
function	$x1 = x^\sigma,\ y1 = y^\sigma,\ z1 = z^\sigma$		0	3	3	15
	$A = x1 + y1 + z1$	2			2	2
	$u = A^\rho$			1	1	5
		2	0	4	6	22
gradient	$x2 = x1/x,\ y2 = y1/y,\ z2 = z1/z,$		3		3	3
	$A1 = \rho\ \sigma\ u/A$		3		3	3
	$u_x = x2\ A1$		1		1	1
	$u_y = y2\ A1$		1		1	1
	$u_z = z2\ A1$		1		1	1
grad. cost					9	9
					15	31

- Two kinds of gains
  - Fewer operations: Theorem: (Griewank) For an n-dimensional function f:
    - Cost (Jacobian) $<$ 5 Cost (f)
    - Cost (Hessian) $<$ 5 n Cost (f)
  - Less use of expensive operations: power (~10 adds), exponential (~5 adds), log (~10 adds), etc.

- Current applications and implications
  - Use Newton methods (and discard DFP, BFGS, and BHHH)
    - AD is now incorporated into much software; AMPL and GAMS, but not empirical packages!
    - L-B-J: already exploits sparseness, could exploit AD
    - Solve stochastic dynamic games
      - Pakes-Maguire and others use Gauss–Seidel methods - sssslooooooowwwwww
      - Ferris-Judd-Schmedders: 40,000 states, 240,000 unknowns; done in 5 minutes on a laptop
  - Perturbation methods
    - Judd, Guu, Gaspar, Anderson, Juillard, Collard, Kim-Kim, Jesus Fernandez-Villaverde, Juan Rubio.
    - Some have incorporated AD ideas into their code: Anderson-Levin-Swanson
    - Laplace expansions in statistics.

Math Tool II: Efficient Function Approximation

- Linear polynomial methods:

$$f(x, y, z, ...) = \sum_{i=1}^{m} a_i \phi_i (x, y, z, ...), \; \phi_i \text{ multivariate polynomials}$$

  – Choices for ϕ are tensor versus complete:

	degree 1 in each variable	degree 2 in each variable
one D	$1, x$	$1, x, x^2$
2D tensor product	$\{1, x\} \otimes \{1, y\}$ $= \{1, x, y, xy\}$	$\left\{1, x, x^2\right\} \otimes \left\{1, y, y^2\right\}$ $= \left\{1, x, x^2, y, y^2, xy, x^2y, xy^2, x^2y^2\right\}$
3D tensor product	$\{1, x\} \otimes \{1, y\} \otimes \{1, z\}$ $= \{1, x, y, z, xy, xz, yz, xyz\}$	
2D complete	$1, x, y$	$1, x, x^2, y, y^2, xy$
3D complete	$1, x, y, z$	$1, x, y, z, xy, xz, yz, x^2, y^2, z^2,$

– Proper notion of "degree" in multivariate context is sum of powers

$$\text{degree}\left(x^i y^j z^k\right) = i + j + k$$

– Complete polynomials like

$$\sum_{i+j+k \leq m} a_{ijk} x^i y^j z^k$$

have far fewer terms than tensor products like

$$\sum_{i=0}^{m} \sum_{j=0}^{m} \sum_{k=0}^{m} a_{ij} x^i y^j z^k$$

with ratio being about $d!$ in d-dimensional case.

– Complete polynomials are better in terms of approximation power per term

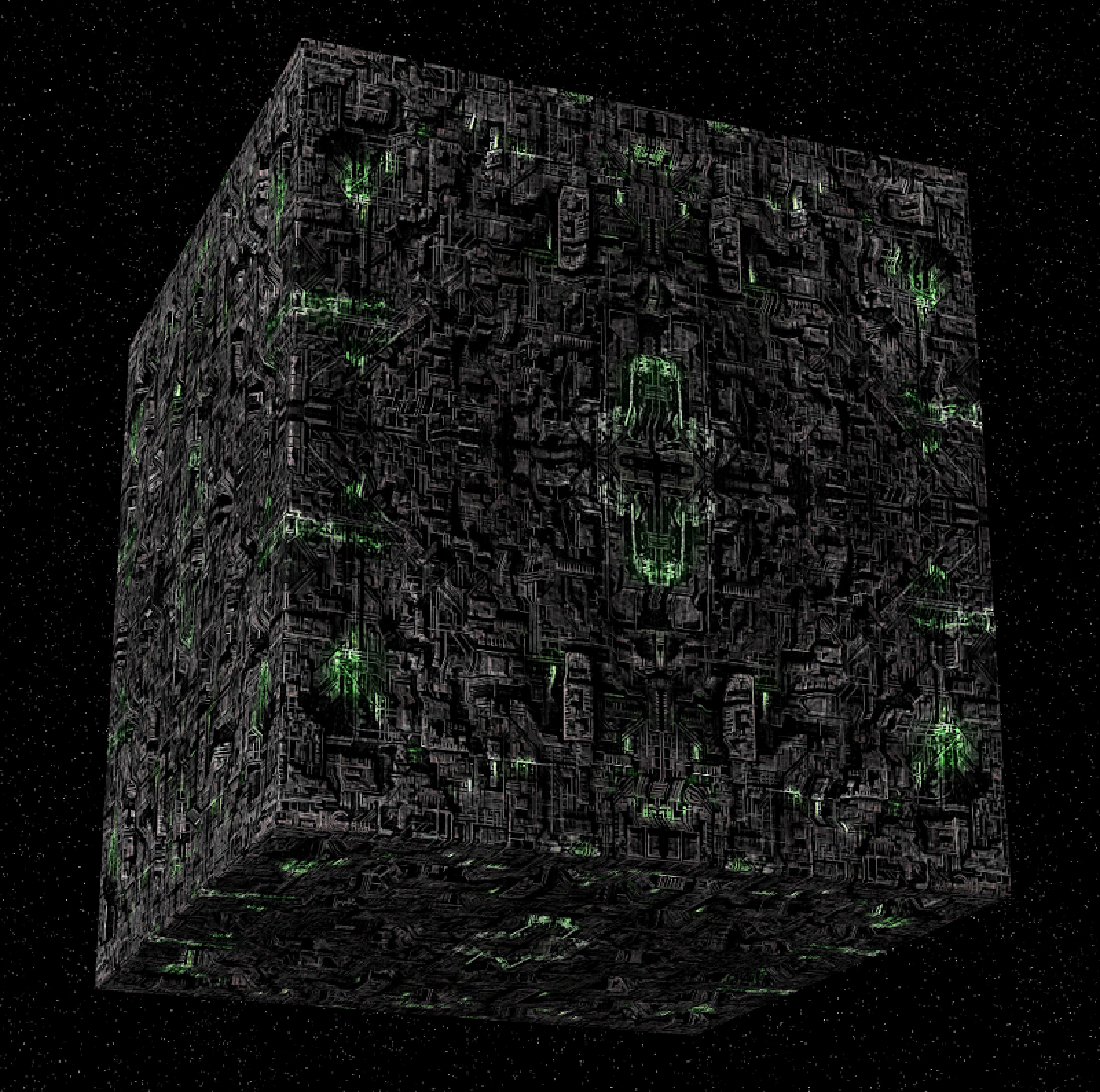
degree k	Number terms in complete poly	Number terms in tensor product
2	$\approx \frac{1}{2} n^2$	3^n
3	$\approx \frac{1}{6} n^3$	4^n

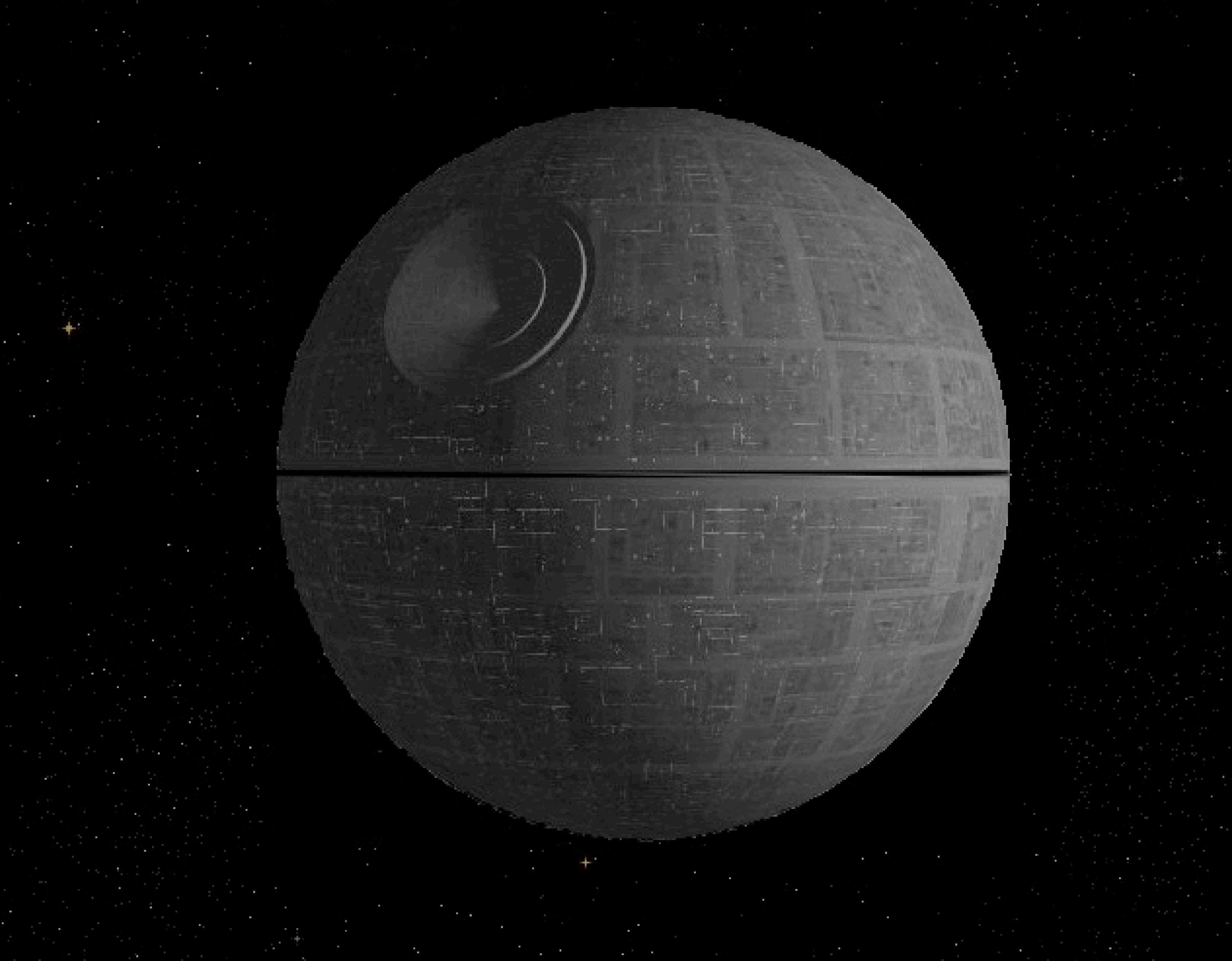
– See Gaspar-Judd (1997)

- Smolyak points and sparse grids
  - Efficient way to approximate smooth high dimensional functions
  - Krueger-Kubler found them to be very effective in stochastic OLG
  - Mertens used them to solve five-D option pricing problem
  - Judd and Mertens are applying them to Bayesian econometrics

Math Tool III: Define the Domain Efficiently

- Choosing the domain of our problem (e.g., states in a DP or dynamic GE model) is important
  - Want to include values for state that are part of the solution
  - Choosing too large a domain will create unnecessary computational burdens.
- More choices with higher dimensions
  - One dimension: Domain is interval; just need to know max and min
  - Two dimensions: More choices - square/rectangle, sphere/ellipse, simplex, etc.
  - Three dimensions: More choices - cube, sphere, ellipsoid, cylinder, simplex, etc.

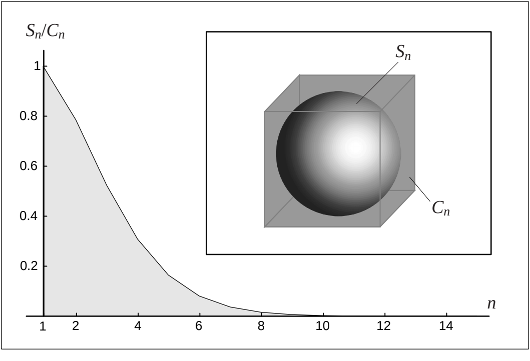




- Cube versus Sphere
  - Spheres are much more compact:
    - In cube of unit length at edge, length of longest diagonal is $n^{1/2}$
    - Ratio of sphere to cube volume is

$$\begin{array}{ll} \frac{\pi^{n/2}}{(n/2)!} & \text{, n even} \\ \frac{2^{n/2+1}\pi^{n/2}}{1\cdot 3\cdot 5\cdot \ldots \cdot n} & \text{, n odd} \end{array}$$

    - Smaller volume reduces costs of approximation; allows one to exploit periodicity
    - Smaller volume reduces cost of integration
  - If solution has a central tendency, then it rarely visits vertices
  - Mathematicians are developing methods for spheres: orthogonal polynomials for hyperspherical coordinates, quadrature rules for spheres


S_n/C_n
1
0.8
0.6
0.4
0.2
S_n
C_n
n
1
2
4
6
8
10
12
14

Math Tool IV: Use Efficient Integration Methods

- New research direction I: Find rules that are good for many polynomials
  - Choose points z_i and weights ω_i, $i = 1, .., m$, to create a quadrature rule,

$$Q(f; z, \omega) = \sum_{i=1}^{n} \omega_i f(z_i)$$

    to minimize errors.

    * The literature is for one-dimensional problems:

$$\min_{z,\omega} \sum_{i=0}^{\infty} \left(Q\left(x^i; z, \omega\right) - \int x^i \, dx \right)^2$$

    * A few mathematicians do this - Gismalla, Cohen, Minka
    * This is not done often since "you can't publish the results".

– I created one for a two-D sphere:

  * We need new formulas if we switch to spheres
  * Choose 12 points (24 coordinates and 12 weights) to minimize sum of squared errors of formula applied to $x^i y^j$, $i, j \leq 20$.
  * Use unconstrained optimization software; use many restarts to avoid local solution
  * Result was

$$\begin{aligned}
&0.2227\ (f[-0.8871, 0] + f[0, -0.8871] + f[0, 0.8871] + f[0.8871, 0]) \\
+&0.2735\ (f[-0.6149, 0.6149] + f[-0.6149, -0.6149] \\
&+f[0.6149, -0.6149] + f[0.6149, 0.61496]) \\
+&1.0744f[-0.3628, 0] + 1.0744f[0, 0.3628] \\
&+1.0744f[0, -0.3628] + 1.0744f[0.3628, 0]
\end{aligned}$$

    with relativized errors of 10(-5) on average and 10(-4) at worst on degree 20 polynomials
  * Result had interesting symmetry - 3 groups of 4 points lying on 3 circles - which gives indication as to what symmetries I should try in higher dimensions.

– General strategy: Look for formulas with small numbers of points to find desirable patterns for point sets, then assume those patterns when searching for bigger formulas.

– General principal: Use your time to come up with ideas, and use the computer to do the tedious work.

  * Idea here: use formulas that integrate an important set of polynomials.
  * Tedious work here: searching for optimal rule that satisfies the criterion.

- New research direction II: Use more information
  - Gauss-Turan methods use derivatives

$$\int_{-1}^{1} f(x)\, dx = \sum_{i=1}^{n} \omega_{i,0} f(z_i) + \sum_{i=1}^{n} \omega_{i,1} f'(z_i) + \sum_{i=1}^{n} \omega_{i,2} f''(z_i)$$

    * n-point formula has $4n$ parameters, and uses $3n$ bits of information to integrate first $4n$ polynomials
    * In one dimension, the cost of f and first two derivatives is about same as three f's, so no gain in one dimension.

– However, Gauss-Turan has potential for high-dimensional integrals

* The formula

$$\int_{-1}^{1}\int_{-1}^{1} f(x,y)\,dx\ dy = \sum_{i=1}^{n} \omega_{i,0} f(x_i,y_i) + \sum_{i=1}^{n} \left(\omega_{i,x} f_x(x_i,y_i) + \omega_{i,y} f_y(x_i,y_i)\right) + \sum_{i=1}^{n} \left(\omega_{i,xx} f_{xx}(x_i,y_i) + \omega_{i,xy} f_{xy}(x_i,y_i) + \omega_{i,yy} f_{yy}(x_i,y_i)\right)$$

  uses $6n$ bits of information at n points - one f evaluation and five derivatives - has $7n$ parameters and can integrate first $7n$ polynomials

* Using automatic differentiation, multidimensional Gauss-Turan will beat regular quadrature rules that use only f values

- Quasi-Monte Carlo (qMC) Methods
  - Sampling methods (including MC) use sequence x_i and computes N-point approximation
$$\int_0^1 f(x)\,dx \equiv \frac{1}{N}\sum_{i=1}^{N} f(x_i) \qquad (1)$$
  - Two simple qMC examples in R^d are
$$\begin{aligned} &\text{Weyl:} && x^n = \left(n\,p_1^{1/2}, \cdots, n\,p_d^{1/2}\right) \bmod 1 \\ &\text{Niederreiter:} && x^n = \left(n\,2^{1/(d+1)}, \cdots, n\,2^{d/(d+1)}\right) \bmod 1 \end{aligned}$$
  - Practical facts
    * Convergence for smooth integrals using Weyl or Neiderreiter is N^{-1}. Others are better.
    * qMC is excellent for high-D (e.g., 360) problems in option pricing problems
  - News:
    * New sequences: randomized $(t-m-s)$ sequences have $N^{-3/2}$ convergence - Owen
    * qMC beats GHK by 10-100 for dimension < 10; beaths GHK by >2 for dimensions 10-50.
    * qMC beats MCMC by 10-100 on ordinary problems - Tribble and Owen.

Math Tool V: Functional Analysis

- Economics problems often reduce to finding unknown functions defined by functional equations
  - Dynamic programming: contraction fixed point map on space of bounded functions using L_∞ norm.
  - Dynamic games
- Functional analysis tells us how to generalize calculus (e.g., Taylor series, IFT, etc.) to spaces of functions

- Can use IFT in Banach spaces (using C^2 topology) to solve dynamic game
  - Hyperbolic discounting
    - Krusell-Kuruscu-Smith (2002) used high-order conjectural variation approach; produced many solutions
    - Judd (2005) used Banach space IFT to prove local existence and uniqueness, and demonstrated nonlocal validity of expansion
  - Stochastic growth model
    - Start with deterministic model to get, e.g., $C(k)$ for $k \in [k0, k1]$
    - Add uncertainty - σ; compute the function $C_\sigma(k), C_{\sigma\sigma}(k), C_{\sigma\sigma\sigma}(k), C_{\sigma\sigma\sigma\sigma}(k)$, etc., functions for $k \in [k0, k1]$
    - Construct series expansion: $\sum_{i=0}^{n} \frac{\sigma^i}{i!} C_{\sigma^i}(k)$.
  - General idea:
    - Differentiate in Banach spaces to derive equations satisfied by $f_\varepsilon(k), f_{\varepsilon\varepsilon}(k)$, $f_{\varepsilon\varepsilon\varepsilon}(k)$, etc., then solve for $f_\varepsilon(k), f_{\varepsilon\varepsilon}(k), f_{\varepsilon\varepsilon\varepsilon}(k)$
    - "Approximate the derivatives"; similar to, but generally better, than "differentiating the approximations"

MAKE MODELLING CHOICES

TO PRODUCE MANAGEABLE MATHEMATICAL PROBLEMS

- Many modelling choices are not essential for the economics.
- Economists should make otherwise inessential choices that reduce computational problems

Modelling Suggestion I: Use Continuous Time

- We pay a high price when we choose discrete-time formulations
- Dynamic programming: "next period's value"
  - Discrete-time: $V(F(x, U(x)))$ - double composition
  - Continuous-time: $V'(x) F(x, U(x))$ - single composition plus multiplication and gradient
  - Stochastic discrete time: $E\{V(F(x_t, U(x_t)), \theta_{t+1}) | \theta_t\}$ - double composition plus multidimensional integral
  - Stochastic continuous time: $V'(x) F(x, U(x)) + \sigma^2 V''(x)$ - single composition, multiplication, gradient, and Hessian
  - Composition of unknown functions (V and U) is far costlier than derivatives for both perturbation and projection methods
- Stochastic games: Doraszelski-Judd show that continuous-time games are orders of magnitude faster than discrete-time games.

Modelling Suggestion II: Use Finite-Dimensional States

- Many economists have problems with infinite-dimensional states, such as the distribution of income
- Alternative approach: There is only a finite number of people
- Example: suppose you have dynamic programming problem with N factories, each with DRTS, with adjustment costs for investment.
  - Bellman equation

$$\begin{aligned} V(k) &= \max_I u(c) + \beta V(k+I) \\ c &= \Sigma_i f(k_i) - \Sigma_i g^i\left(I^i(k)\right) \end{aligned}$$

  - Equations defining $V(k)$ and $I(k)$:

$$\begin{aligned} V(k) &= u(c) + \beta V(k + I(k)) \\ 0 &= -u'(c)\left(1 + \alpha I^i(k)\right) + V_i(k + I(k)) \end{aligned}$$

  - Idea: Use perturbation method to compute Taylor series for $V(k)$ and the $I^i(k)$

- Problems:
  * V_i is a vector of length N; V_{ij} is a matrix with N^2 elements;
  * $I^i(k)$ is a list of N functions; $I^i_j(k)$ is an $N \times N$ matrix; $I^i_{jm}(k)$ is $N \times N \times N$ tensor, etc.
  * If $N = 10^9$, that is a lot of unknowns
- Solution: Exploit symmetry at steady state
  * $V_i = V_j, \forall i, j$
  * $V_{ii} = V_{11}, \forall i; V_{ij} = V_{12}, \forall i \neq j$
  * $V_{iii} = V_{111}, \forall i; V_{iij} = V_{112}, V_{ijj} = V_{122}, \forall i \neq j; V_{ijm} = V_{123}, \forall i \neq j \neq m \neq i$
  * Similarly for I^i functions
- High-order Taylor series are feasible
  * The number of unknowns when computing q'th derivative is $2q$ independent
  * Solutions depend on N; take $N \rightarrow \infty$ to find infinite population solution
  * Risk - idiosyncratic and aggregate - can be added with little extra computational cost.
- Similar to Gaspar-Judd (1997) use of symmetry, but far more efficient

Modelling Suggestion III: Get Rid of Kinks

- General observation: the more smoothness, the better for computation.
- Economists love to put in discontinuities. For example, Hubbard and Judd (1986)
  - Wanted to examine tax policy implications of borrowing constraints.
  - Assumed one could not borrow against future wages; equivalent to

$$r(W) = \begin{cases} r, W > 0 \\ \infty W < 0 \end{cases}$$

    or, equivalently,

$$u(c, W) = \begin{cases} u(c), W > 0 \\ -\infty \ W < 0 \end{cases}$$

  - Results were interesting, but hampered by computational inefficiencies

- Is this economically reasonable? Borrowing is not infinitely painful
  - First, go to parents.
  - Second, run up credit card debt.
  - In general, there is a set of sources of credit, with rising interest rates
  - Empirical fact: people do have debt!
- General point: kinks and discontinuities create problems but there are few problems where nonsmooth functions are necessary

FUTURE TOOLS

- Economists should not just think in terms of the hardware and software available today.
- Some new tools will be particularly valuable for solving high dimensional models.

Experimental Mathematics

- MC methods as practiced is very useful and sound but not supported by usual mathematical theorems. Real proof is
  - Suppose $f(x) = \sum_{i=0}^{\infty} a_i x^i$ on $[0,1]$ and $\sum_{i=K}^{\infty} a_i x^i$ is negligible for some K
  - Suppose computations show that a sequence X_i properly computes $\int x^i dx$ at rate $N^{-1/2}$ for each $i < K$.
  - Then, MC will compute $\int f(x)$ at rate $N^{-1/2}$
- This is experimental mathematics *NOT* probability theory!
- Experimental math:
  - Test out conjecture on many cases to explore validity
  - Combine computational results with pure math to arrive at conclusions with known range of validity
  - Computational results may inspire theorems, such as Neiderreiter analysis of LCM.

- Problem is not with using MC, but with understanding logical underpinnings.
  - MC in practice is not based on probability theory
  - It is *inspired* by probability theory, but theorems do not apply
  - This inspiration led to search for pMC sequences which, by *testing*, were found to do a good job on some problems
- Why are these logical points important?
  - All agree that Monte Carlo is a very important and useful tool.
  - Recognition of the true foundation for MC will encourage us to develop other methods based on a similarly disciplined combination of analysis and computational experimentation.

Computing Speed

- We need more speed to do the necessary heavy lifting - searches for good methods, symbolic manipulation, experimental mathematics - implicit in the ideas mentioned above.
- More speed is coming
  - Massively parallel architectures - 10^5 processors in next BlueGene
  - Network computing
  - Multicore processors on desktops.

Griebel-Wozniakowski Theorem

- Question: Are there good rules out there to defeat the curse of dimensionality?
- Answer: Yes, if we formulate problem in reasonable spaces.
- "On the Optimal Convergence Rate of Universal and Non-Universal Algorithms for Multivariate Integrationand Approximation" by Griebel and Wozniakowski
  - Consider functions in reproducing kernel Hilbert spaces.
    * If the kernel is a product of univariate kernels, i.e.,

      $$\int_{[0,1]^n} g(x)\,dF(x) = \int_{[0,1]} \cdots \int_{[0,1]} \int_{[0,1]} g(x)\,dF_1(x_1)\,dF_2(x_2)\ldots dF_n(x_n)$$

      then optimal algorithm is as fast as slowest optimal algorithm of univariate kernels
    * Hence, the optimal rate of convergence of universal algorithms for product kernels does not depend in dimension!
  - Proof is nonconstructive, but tells us that computer searches are not necessarily futile.
- Economics problems generally fit this description

Quantum Computing

- New technology may break curse of dimensionality
- Quantum computer example
  - Load quantum computer with a function f and a number n.
  - ZAP it and it becomes n computers (more precisely, the quantum state of the computer will be a superposition of the n possible states) where computer i computes $f(i)$, $i = 1, .., n$
  - ZAP it $n^{-1/2}$ times
  - Take a random draw among the n computers before they collapse back to one, but sample is now biased so that you get $\max f(i)$ with probability $1 - n^{-1}$!

- Quantum complexity theory
  - Examines possible efficiency of quantum computer algorithms.
  - There are examples of where quantum computing breaks curse of dimensionlity.
  - "Path Integration on a Quantum Computer," Traub and Wozniakowski (2001).
    * Path integration on a quantum computer is tractable - i.e., no curse of dimensionality.
    * Path integration on a quantum computer can be solved roughly $\varepsilon^{-1/2}$ times faster than on a classical computer using randomization
    * The number of quantum queries is the square root of the number of function values needed on a classical computer using randomization.
  - In general, integration is faster on a quantum computer than a classical computer - Brassard-Hoya-Mosca-Tapp.

CONCLUSION

- If you formulate models in the right way, and If you use best available math, then you can avoid the curse of dimensionality
- New developments are making that easier to do.
- The path is clear, but there is a lot of work to do to build the road.