Slow Growth for Sparse Grids

John Burkardt, Clayton Webster, Guannan Zhang

.....

http://people.sc.fsu.edu/~jburkardt/presentations/...
slow_growth_2014_savannah.pdf

.....

SIAM UQ Conference
31 March - 03 April 2014

References:

- Florian Heiss, Viktor Winschel,
  Likelihood approximation by numerical integration on sparse grids,
  Journal of Econometrics, Volume 144, 2008, pages 62-80.
  Matlab program for sparse grid generation with slow growth
- Erich Novak, Klaus Ritter,
  Simple cubature formulas with high polynomial exactness,
  Constructive Approximation, Volume 15, Number 4, December 1999, pages 499-522.
  Exactness constraint for sparse grids
- Knut Petras,
  Smolyak Cubature of Given Polynomial Degree with Few Nodes for Increasing Dimension, Numerische Mathematik, Volume 93, Number 4, February 2003, pages 729-753.
  C program for sparse grid generation with slow growth
- Miroslav Stoyanov,
  User Manual: TASMANIAN Sparse Grids,
  ORNL Report, Oak Ridge National Laboratory, 2013.
  C++ program for sparse grid generation with slow growth

SLOW GROWTH FOR SPARSE GRIDS

- **Introduction**
- Clenshaw-Curtis
- Gauss-Legendre
- Gauss-Patterson
- Conclusion

Need to Estimate Multidimensional Integrals

A vital task in uncertainty quantification estimates integrals of the form

$$I(f)(x) = \int_{\Omega} f(x, \omega)\, \rho(\omega)\, d\omega$$

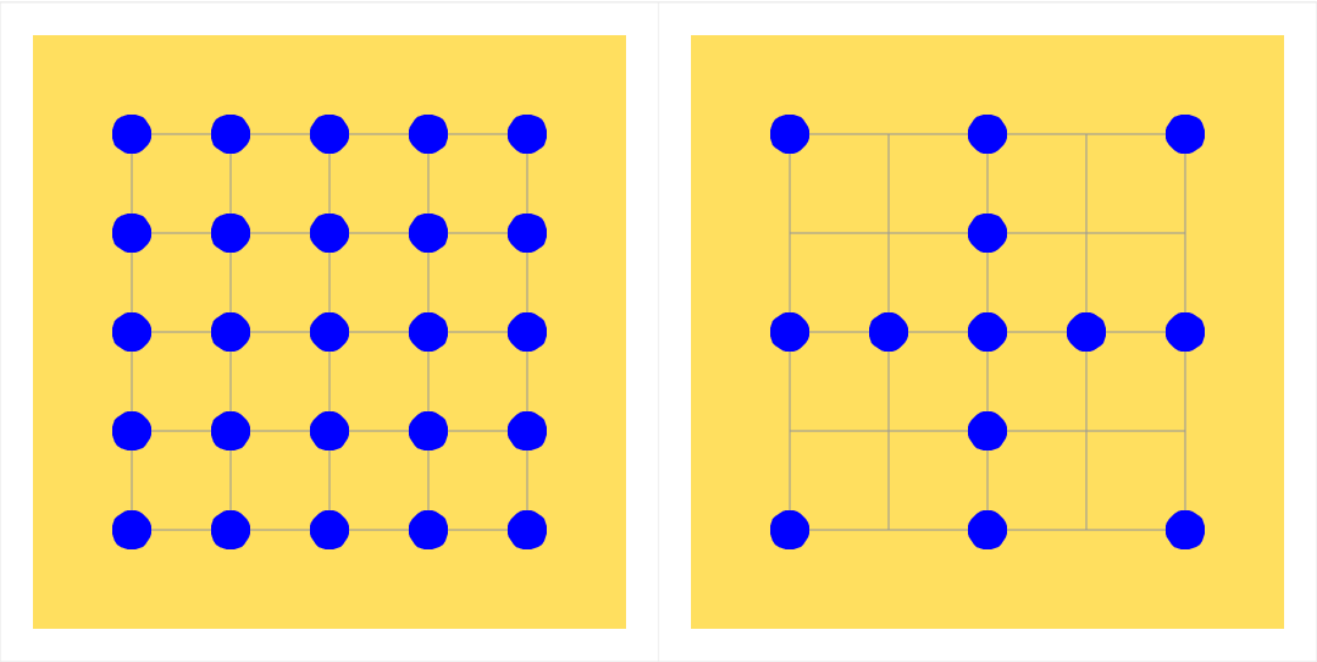
where the uncertainty parameter vector ω lies in Ω, the d-dimensional unit hypercube, or $\mathbb{R}^d$ or some general product region.

To estimate $I(f)$, we may choose a family of product rules with an index ℓ, such that $Q_\ell(f)$ has a known exactness $p(\ell)$, so that using successive elements of the family gives us estimates of $I(f)$ and the error.

However, if the 1D rule requires n points to achieve an exactness of p, the corresponding product rule requires n^d points, ruling out even moderate values of n and d.

Sparse Grids Can Beat Product Grids

A product rule of exactness p actually catches all integrands of **maximum** degree p, when only **total** degree terms are needed. In high dimensions, this represents a substantial inefficiency. Sparse grids seek to match, but not exceed, the exactness requirement by combining low order product rules.



Here, the total exactness $p = 5$ of the product grid on the left (n=25 points) is matched by that of the sparse grid on the right (n=13 points).

Sparse Grids Can Be Very Efficient

The classic sparse grid family is constructed from a carefully selected set of nested 1D Clenshaw-Curtis rules for the interval [-1,+1]; nesting requires that the order of the rules increases exponentially: 1, 3, 5, 9, 17, 33, 65, ...

Suppose we use these factors to build a product rule, and a sparse grid, for the 5-dimensional unit hypercube $[-1,+1]^5$.

The table indicates the number of points necessary to achieve successive exactness levels:

p:	1	2	3	4	5	6	7	8	9	10
n(product)	1	32	243	1024	3125	7776	16807	32768	59049	100000
n(sparse)	1	-	11	-	29	-	65	-	145	-

Despite the fact that the 1D CC rules increase exponentially in size, the resulting sparse grid grows far more slowly than the product rule.

The Cost of a Bad Start

Now fix an exactness level p, and allow the dimension to increase.

Here are the number of points n needed to achieve a polynomial exactness $p = 9$ in various dimensions:

d:	1	2	3	4	5	6	7	8	9	10
n(product)	9	81	729	6K	59K	531K	4M	43M	387M	3B
n(sparse)	33	145	441	1K	2K	4K	9K	15K	26K	41K

On each step, the product rule order increases by a factor of 9, while the sparse grid factor growth factor is dropping below 2.

Again, we (eventually) have a substantial benefit, but it looks like we suffer initially because of a bad starting value. Do we really need 33 points in 1D to get exactness $p = 9$? Of course not - a 9 point rule would do it.

This peculiar value is an artifact of the way Clenshaw-Curtis sparse grids are defined, which we usually ignore because we never look closely at the 1D case.

But perhaps we should!

SLOW GROWTH FOR SPARSE GRIDS

- Introduction
- **Clenshaw-Curtis**
- Gauss-Legendre
- Gauss-Patterson
- Conclusion

Sparse Grids in 1D

Once we have specified a index list of 1D quadrature rules or "factors", Smolyak allows us to generate a sparse grid in any dimension.

If we set up the Smolyak machinery, and ask it to generate a "sparse grid" in 1D, then we get back the original 1D quadrature rules.

It is common to expect a sparse grid of level ℓ to have an exactness that grows linearly with the level:

$$p = 2\ell + 1 \text{ (Novak \& Ritter)}$$

What can we say about our peculiar 1D Clenshaw-Curtis "sparse grid"?

ℓ = level	0	1	2	3	4	5	6	7	8	9	10	...
n = points	1	3	5	9	17	33	65	129	257	513	1025	...
p = exactness	1	3	5	9	17	33	65	129	257	513	1025	...
p(necessary)	1	3	5	7	9	11	13	15	17	19	21	...

For the 1D sparse grid, our order and exactness grow exponentially:

$$n = 2^\ell + 1, \quad 1 \leq \ell$$

$$p = 2^\ell + 1 = n$$

Nesting

Why are we using a 1D factor family that grows exponentially?

We are trying to control point growth in the multidimensional rule. Recall that the points of a sparse grid are the logical sum of the points of a collection of product grids that satisfy a constraint on their definition.

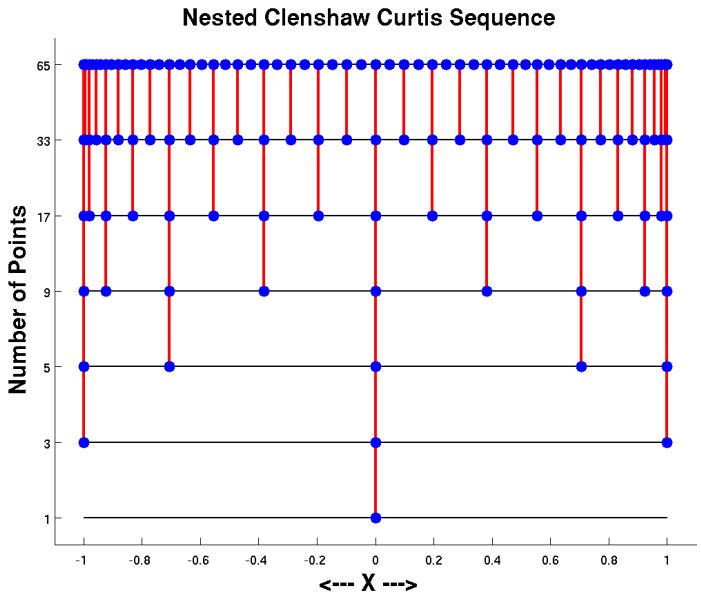
If all these product rules are defined using a 1D **nested** family, then when we gather together the logical sum of the product grids, the total number of points can be greatly reduced.

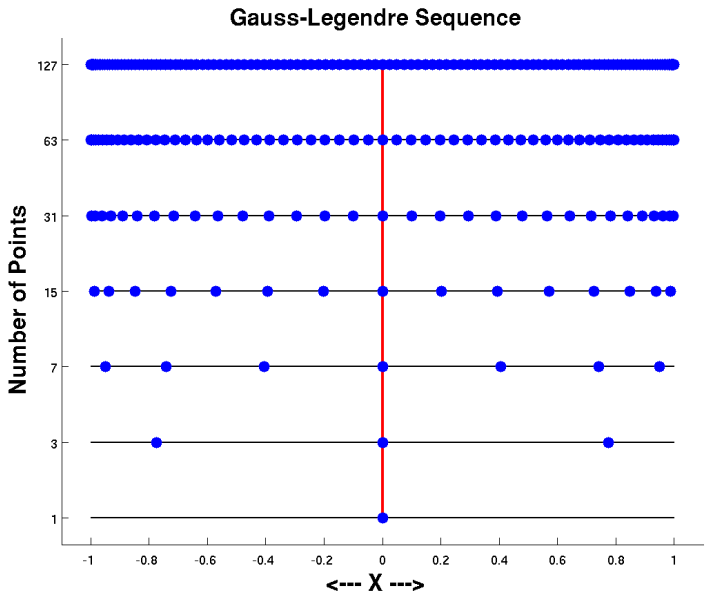
Compare in 2D the nested CCE versus the non-nested GLE (Gauss-Legendre exponential) sparse grids.

$\ell =$ level	0	1	2	3	4	5	6	7	8	9	...
n (CCE)	1	5	13	29	65	145	321	705	1537	3329	...
n (GLE)	1	5	22	75	224	613	1578	3887	9268	21561	...

Nesting is a very powerful technique to keep sparse grids "affordable"!

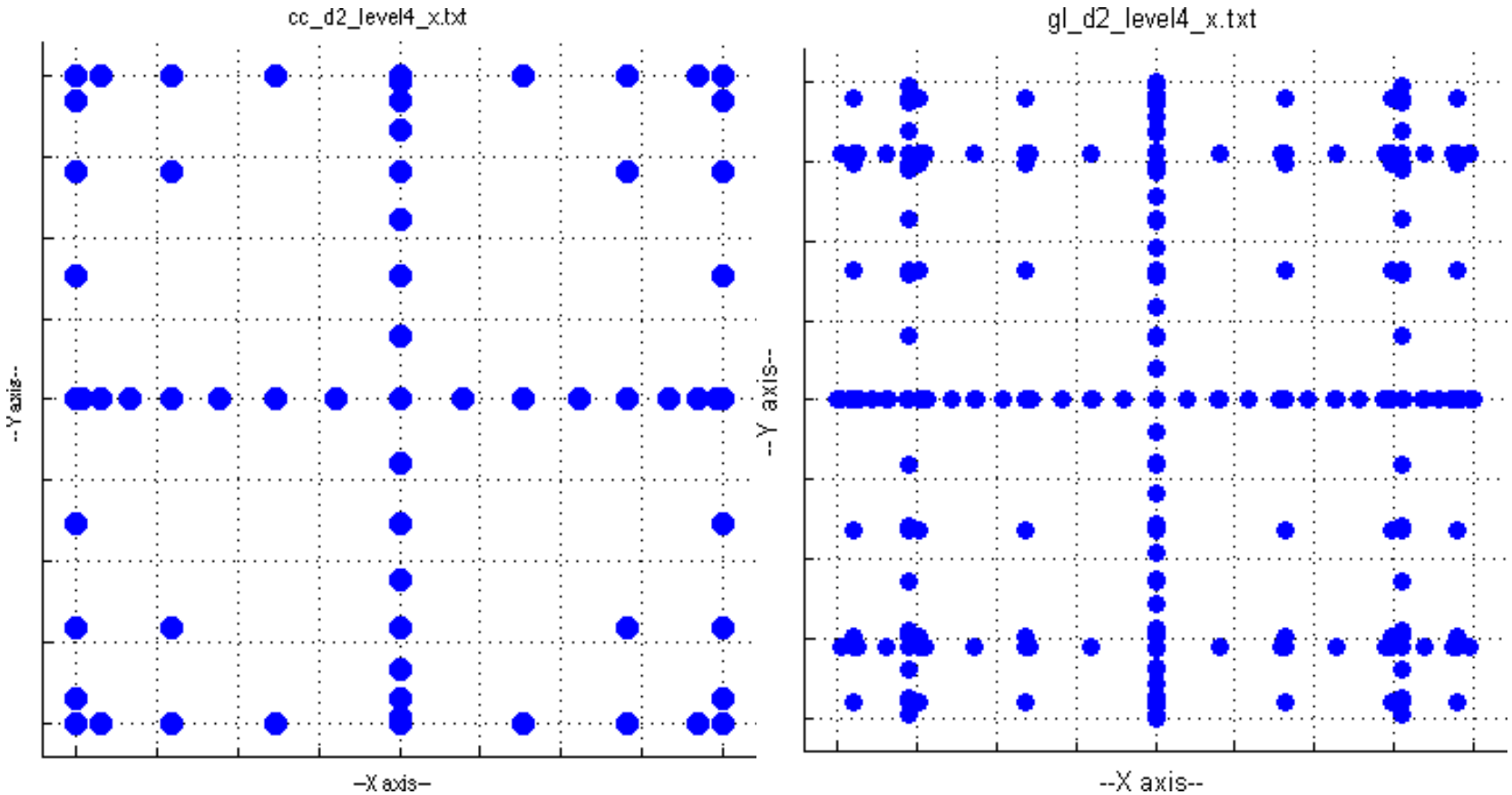
Nested CCE family / Nonnested GLE family





The CCE family is completely nested
in the GLE family, only the 0.0 value is repeated. Moreove, because it is an open rule, the size of each 1D factor is about double that of CCE.

Nesting in 2D Sparse Grids



Nesting keeps the Clenshaw Curtis sparse grid efficient (65 points). The Gauss-Legendre sparse grid has 224 distinct points.

Can We Abandon Nesting?

One alternative to the exponentially growing version of the CC rule would be to use a Clenshaw-Curtis family of odd orders and linear growth, $n = 1, 3, 5, 7, 9, ...$, which will exactly meet the Novak & Ritter exactness requirement.

This family is not nested. So our tradeoff is that our sparse grids will be combining product rules of lower order, but with more distinct points.

What is the effect in 2D?

ℓ = level	0	1	2	3	4	5	6	7	8	9	...
n (CCE)	1	5	13	29	65	145	321	705	1537	3329	...
n (CCL)	1	5	13	29	57	105	177	281	425	611	...

The CCL rule doesn't show an advantage until the underlying factors begin to differ, after which we see a big reduction.

Does this 2D result carry over to higher dimensions?

Keep Nesting, Slow Exponentiation

A second alternative would be to retain the exponentially growing factor family, but to always use the lowest such rule that will satisfy the exactness requirement.

In other words, we start with the CCE factor family $n = 1, 3, 5, 9, 17, 33...$, but repeat rules where possible.

Compare the CCE, CCL and CCS 1D factor families:

ℓ = level	0	1	2	3	4	5	6	7	8	9	...
p (required)	1	3	5	7	9	11	13	15	17	19	...
n (CCE)	1	3	5	9	17	33	65	125	257	513	...
n (CCL)	1	3	5	7	9	11	13	15	17	19	...
n (CCS)	1	3	5	9	9	17	17	17	17	33	...

The CCS factor family grows faster than CCL, and does so in exponential "jumps" but makes those jumps far less often than the CCE family, and inherits the advantages of nestedness.

Compare CCE, CCL, CCS

If we build a 2D sparse grid from the CCS rule, what happens?

Does the 2D sparse grid inherit the "stutter" of the 1D factors?

ℓ = level	0	1	2	3	4	5	6	7	8	9	...
n (CCE)	1	5	13	29	65	145	321	705	1537	3329	...
n (CCL)	1	5	13	29	57	105	177	281	425	611	...
n (CCS)	1	5	13	29	49	81	129	161	225	257	...

and for 6D:

ℓ = level	0	1	2	3	4	5	6	7	8	9	...
n (CCE)	1	13	85	389	1,457	4,865	15,121	44,689	127,105	350,657	...
n (CCL)	1	13	85	389	1,433	4,533	12,961	33,817	82,153	188,039	...
n (CCS)	1	13	85	389	1,409	4,289	11,473	27,697	61,345	126,401	...

And for 10D:

ℓ = level	0	1	2	3	4	5	6	7	...
n (CCE)	1	21	221	1,581	8,801	41,265	171,425	652,065	...
n (CCL)	1	21	221	1,581	8,761	40,425	162,385	584,665	...
n (CCS)	1	21	221	1,581	8,721	39,665	155,105	536,705	...

As d increases, the CCL and CCS advantages are delayed and decreased.

SLOW GROWTH FOR SPARSE GRIDS

- Introduction
- Clenshaw-Curtis
- **Gauss-Legendre**
- Gauss-Patterson
- Conclusion

GLE Factor Family

Consider now using Gauss-Legendre rules for our factor family.

We begin with the GLE family, of orders 1, 3, 7, 15, ...

Because the Gauss-Legendre rules are open, the growth is faster than for the (closed) Clenshaw Curtis rules.

ℓ = level	0	1	2	3	4	5	6	7	8	9	10	...
n = points	1	3	7	15	31	63	127	255	511	1023	2047	...
p = exactness	1	5	13	29	61	125	253	509	1021	2045	4093	...
p(necessary)	1	3	5	7	9	11	13	15	17	19	21	...

The GLE order is growing exponentially, and at a rate about double that of the CCE family.

$$n(\text{GLE})(\ell) = 2^{\ell+1} - 1$$
$$p(\text{GLE})(\ell) = 2 \cdot (2^{\ell+1} - 1) - 1 = 2 \cdot n(\text{GLE})(\ell) - 1$$

Moreover, the exactness is about 4 times that of the CCE rule, and fantastically exceeds the requirements of the Novak & Ritter constraint.

GLL Factor Family

Using an exponentially growing family for the CC rule was defensible, because of nesting. But the GL family has (almost) no nesting, so there is no reason to use exponential growth.

The analogue of our CCL rule would be a GLL rule that uses the lowest order rule satisfying the Novak & Ritter exactness requirement. Because of the power of GL rules, the GLL sequence would have the orders 1, 2, 3, 4, ...

However, let us consider trying for a tiny bit of nesting, defining the alternative GLO rule, which uses the lowest order odd rule satisfying the constraint.

Here is the order and exactness table for the 1D GLL and GLO factors:

ℓ = level	0	1	2	3	4	5	6	7	8	9	10	...
n(GLE)	1	3	7	15	31	63	127	255	511	1023	2047	...
p(necessary)	1	3	5	7	9	11	13	15	17	19	21	...
n(GLL)	1	2	3	4	5	6	7	8	9	10	11	...
p(GLL)	1	3	5	7	9	11	13	15	17	19	21	...
n(GLO)	1	3	3	5	5	7	7	9	9	11	11	...
p(GLO)	1	3	5	9	9	13	13	17	17	21	21	...

$$n(\mathrm{GLL})(\ell) = 2\ell + 1$$

$$n(\mathrm{GLO})(\ell) = 2 \cdot \lfloor \frac{\ell + 1}{2} \rfloor + 1$$

Point Counts for GLE/GLL/GLO

2D:

ℓ = level	0	1	2	3	4	5	6	7	8	9	...	
n (GLE)	1	5	21	73	221	609	1,573	3,881	9,261	21,553	49,205	...
n (GLL)	1	5	13	29	53	89	137	201	281	381	501	...
n (GLO)	1	5	9	17	29	41	65	81	121	141	201	...

10D:

ℓ = level	0	1	2	3	4	5	6	7	...	
n (GLE)	1	21	261	2,441	18,881	126,925	764,365	4,208,385	21,493,065	...
n (GLL)	1	21	221	1,581	8,761	40,405	162,025	581,385	1,904,465	...
n (GLO)	1	21	201	1,201	5,281	19,165	61,285	177,525	474,885	...

15D:

ℓ = level	0	1	2	3	4	5	6	7	...
n (GLE)	1	31	541	6,911	71,621	635,687	4,995,357	35,537,007	...
n (GLL)	1	31	511	5,921	53,921	409,727	2,695,967	15,751,937	...
n (GLO)	1	31	451	4,151	27,671	145,697	644,937	2,506,137	...

Here, the GLO rule greatly outperforms the GLL rule, and does do by using bigger rules! The secret is that we have gained some nesting opportunities, which turn out to have a tremendous payback.

SLOW GROWTH FOR SPARSE GRIDS

- Introduction
- Clenshaw-Curtis
- Gauss-Legendre
- **Gauss-Patterson**
- Conclusion

The Gauss-Patterson Factor Family

Nesting and the doubled exactness of Gaussian rules are two techniques that have a significant influence on the properties of sparse grids.

This suggests looking at a Gauss-Patterson (GP) factor family.

The GP family begins with the 1 and 3 point GL rules. Thereafter, given a rule with n points, the next rule fixes those points, and adds $n+1$ new points, enforcing nesting. A Gauss procedure squeezes out the best accuracy possible, given the constraint that the old points must not be moved.

The result is a nested family with the same exponential growth as GLE and somewhat reduced exactness,

GPE Factor Family

Here is the exactness table for the GPE 1D factor family:

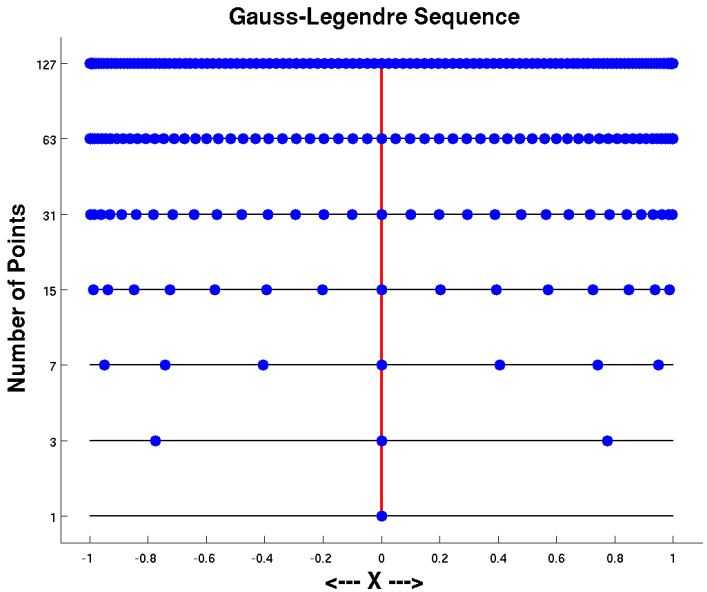
ℓ = level	0	1	2	3	4	5	6	7	8	9	10	...
n = points	1	3	7	15	31	63	127	255	511	1023	2047	...
p = exactness	1	5	11	23	47	95	191	383	767	1535	3071	...
p(necessary)	1	3	5	7	9	11	13	15	17	19	21	...

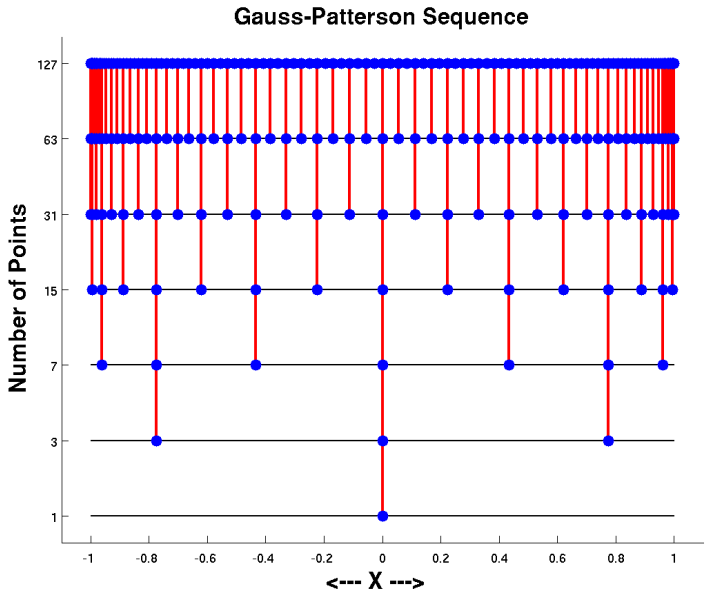
The number of points is the same as for GLE, while the exactness is reduced:

$$n(\text{GPE})(\ell) = 2^{\ell+1} - 1$$

$$p(\text{GPE})(\ell) = 1.5 \cdot (2^{\ell+1} - 1) + 0.5 = 1.5 \cdot n(\text{GPE})(\ell) + 0.5$$

GLE versus GPE





The GLE family is not nested, but the GPE family is, and retains much of the exactness of Gauss rules.

Here is a quick comparison of GLE and GPE in 2D:

ℓ = level	0	1	2	3	4	5	6	7	8	9	10	...
n (GLE)	1	5	21	73	221	609	1,573	3,881	9,261	21,553	49,205	...
n (GPE)	1	5	17	49	129	321	769	1,793	4,097	9,217	20,481	...

GPS: A "Slow" Variant of GPE

Let's go ahead and define a GPS family which only selects the next 1D factor when the Novak & Ritter exactness constraint requires it.

Here are sample point counts comparing GPE and GPS for 2D:

ℓ = level	0	1	2	3	4	5	6	7	8	9	10	...
n (GPE)	1	5	17	49	129	321	769	1,793	4,097	9,217	20,481	...
n (GPS)	1	5	9	17	33	33	65	97	97	161	161	...

and for 6D:

ℓ = level	0	1	2	3	4	5	6	7	8	9	...
n (GPE)	1	13	97	545	2,561	10,625	40,193	141,569	4,710,417	14,960,657	...
n (GPS)	1	13	73	257	737	1,889	4,161	8,481	16,929	30,689	...

and for 10D:

ℓ = level	0	1	2	3	4	5	6	7	8	9	...
n (GPE)	1	21	241	2,001	13,441	77,505	397,825	1,862,145	8,085,505	32,978,945	...
n (GPS)	1	21	201	1,201	5,281	19,105	60,225	169,185	434,145	1,041,185	...

SLOW GROWTH FOR SPARSE GRIDS

- Introduction
- Clenshaw-Curtis
- Gauss-Legendre
- Gauss-Patterson
- **Conclusion**

Compare the Champions

In summary, we have "improved" versions of CCE, GLE and GPE. How do they stack up against each other?

2D:

ℓ = level	0	1	2	3	4	5	6	7	8	9	10	...
n (CCS)	1	5	13	29	49	81	129	161	225	257	385	...
n (GLO)	1	5	9	17	29	41	65	81	121	141	201	...
n (GPS)	1	5	9	17	33	33	65	97	97	161	161	...

10D:

ℓ = level	0	1	2	3	4	5	6	7	8	9	...
n (CCS)	1	21	221	1,581	8,721	39,665	155,105	536,705	1,677,665	4,810,625	...
n (GLO)	1	21	201	1,201	5,281	19,165	61,285	177,525	474,885	1,192,425	...
n (GPS)	1	21	201	1,201	5,281	19,105	60,225	169,185	434,145	1,041,185	...

Conclusion

Sparse grids outperform product rules.

Even sparse grids rapidly grow in cost as level or dimension increases.

The Ritter & Novak exactness constraint tells you the minimum exactness you must achieve. Cut your rule down to that level!

Because a sparse grid is built from many product grids superimposed, nesting is a crucial tool to control the point count.

A Gauss rule can achieve a given exactness with fewer points; but this happens at the expense of nesting.

A Gauss-Patterson factor family combines nesting and moderate exactness, for an efficient sparse grid.

Software implementations appear in **nwspgr** (Heiss & Winschel), **smolpack** (Petras), and **tasmanian** (Stoyanov).