Abstract

Nearly every numerical analysis algorithm has computational complexity that scales exponentially in the underlying physical dimension, a phenomenon dubbed the "Curse of Dimensionality." I will present a method to bypass this curse, based on representing functions of many variables as sums of separable functions. We will first consider what kinds of functions can be well-represented in this way, and what these representations look like. Then we will consider what algorithms are needed to compute using functions in this representation.

Computing in High Dimensions with Sums of Separable Functions

Martin J. Mohlenkamp

Department of Mathematics

in collaboration with
Gregory Beylkin (U. of Colorado)

Goal

Enable you to consider high-dimensional problems from my perspective, and determine if this approach is useful to you.

Not a Goal

Convince you that this method is better than any particular other method for any particular problem.

Overview

- The curse of dimensionality.
- Sums of Separable Functions.
- Examples of separated representations.
- Properties and issues.
- Computational paradigm, strong form.
- Computational paradigm, weak form.

The Curse of Dimensionality

In view of all that we have said in the foregoing sections, the many obstacles we appear to have surmounted, what casts the pall over our victory celebration? It is the curse of dimensionality, a malediction that has plagued the scientist from earliest days.

—Richard Bellman (1961)

The Curse of Dimensionality

If a function in d dimensions needs a discretization of M points per direction, then it needs M^d total points.

M^d is impossibly large for even moderate M and d:
100^{10} =10000000000000000000
100^{20} =100000000000000000000000000000000000000
100^{30} =1000000000000000000000000000000000000000000000000

Implications: In d dimensions

- It is useless to think about functions on regular grids.
- It is useless to think about "general" functions.

But we can't just give up. ¿What do we do?

A Separable Function Avoids the Curse

A separable function in dimension d like

$$f(\mathbf{x}) = f(x_1, \ldots, x_d) = \prod_{i=1}^{d} f_i(x_i)$$

only costs dM to represent, so it avoids the curse.

If the best separable approximation to the function that you want to represent is not good enough, then you need some way to get a better approximation.

A Sum of Separable Functions

A sum of separable functions

$$f(\mathbf{x}) = \sum_{l=1}^{r} s_l \prod_{i=1}^{d} f_i^l(x_i)$$

can approximate better with cost rdM.

If we can get a good approximation with small r, we have defeated the *curse of dimensionality*.

A tensor product basis uses this form, but r grows exponentially with the dimension.

"Sparse grid" methods use this form with smaller r, but still exponentially growing.

Constraints are Devastating

Example: If $\{g_j\}_{j=1}^{2d}$ form an orthonormal set and

$$f(\mathbf{x}) = \prod_{i=1}^{d} g_i(x_i) + \prod_{i=1}^{d} (g_i(x_i) + g_{i+d}(x_i))$$

then an orthogonality constraint would force us to multiply out,

$$\begin{aligned} f(\mathbf{x}) = \prod_{i=1}^{d} g_i(x_i) \qquad & + g_1(x_1) \prod_{i=2}^{d} (g_i(x_i) + g_{i+d}(x_i)) \\ & + g_{d+1}(x_{d+1}) \prod_{i=2}^{d} (g_i(x_i) + g_{i+d}(x_i)) \\ = \cdots & \end{aligned}$$

and have $r = 2^d$ instead of $r = 2$.

Unconstrained!!!

The key to keeping r small is removing constraints such as orthogonality from the f_i^l.

The unconstrained problem is nonlinear, and so harder to do.

There is a much richer structure to sums of separable functions than one would suppose. It is poorly understood.

The Separated Representation of the World

Extend the notion of "sum-of-separable" to:

(Linear) Operators: $\mathcal{L} = \sum_{l=1}^{r} s_l \bigotimes_{i=1}^{d} \mathcal{L}_i^l$

Vectors: $\mathbf{V} = V(j_1, \ldots, j_d) \approx \sum_{l=1}^{r} s_l \prod_{i=1}^{d} V_i^l(j_i) = \sum_{l=1}^{r} s_l \bigotimes_{i=1}^{d} \mathbf{V}_i^l$

Matrices:

$$\mathbb{M} = M(j_1, j_1'; \ldots; j_d, j_d') \approx \sum_{l=1}^{r} s_l \prod_{i=1}^{d} M_i^l(j_i, j_i') = \sum_{l=1}^{r} s_l \bigotimes_{i=1}^{d} \mathbb{M}_i^l$$

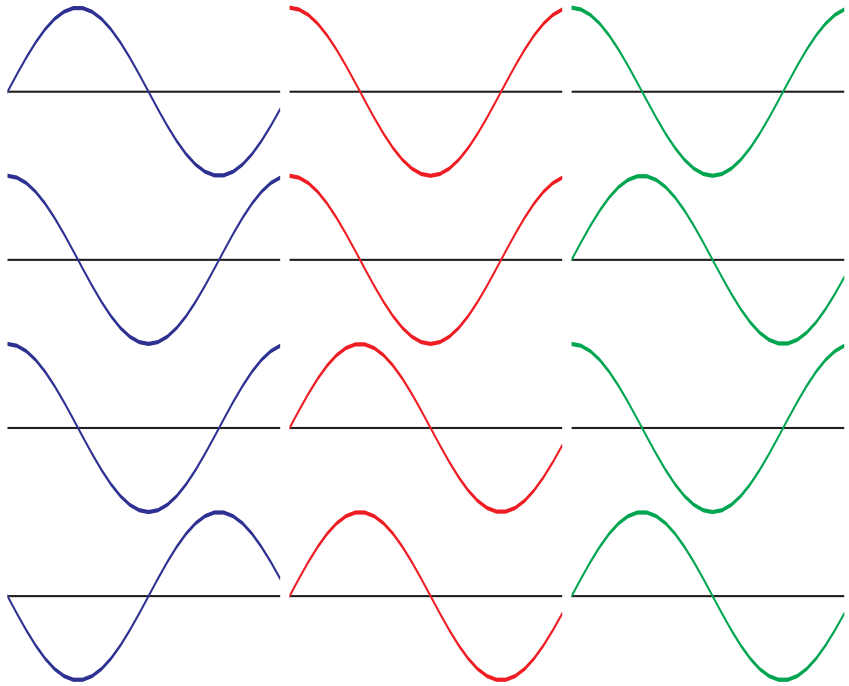
We call r the *separation rank*.

Example: Sine of the sum of several variables

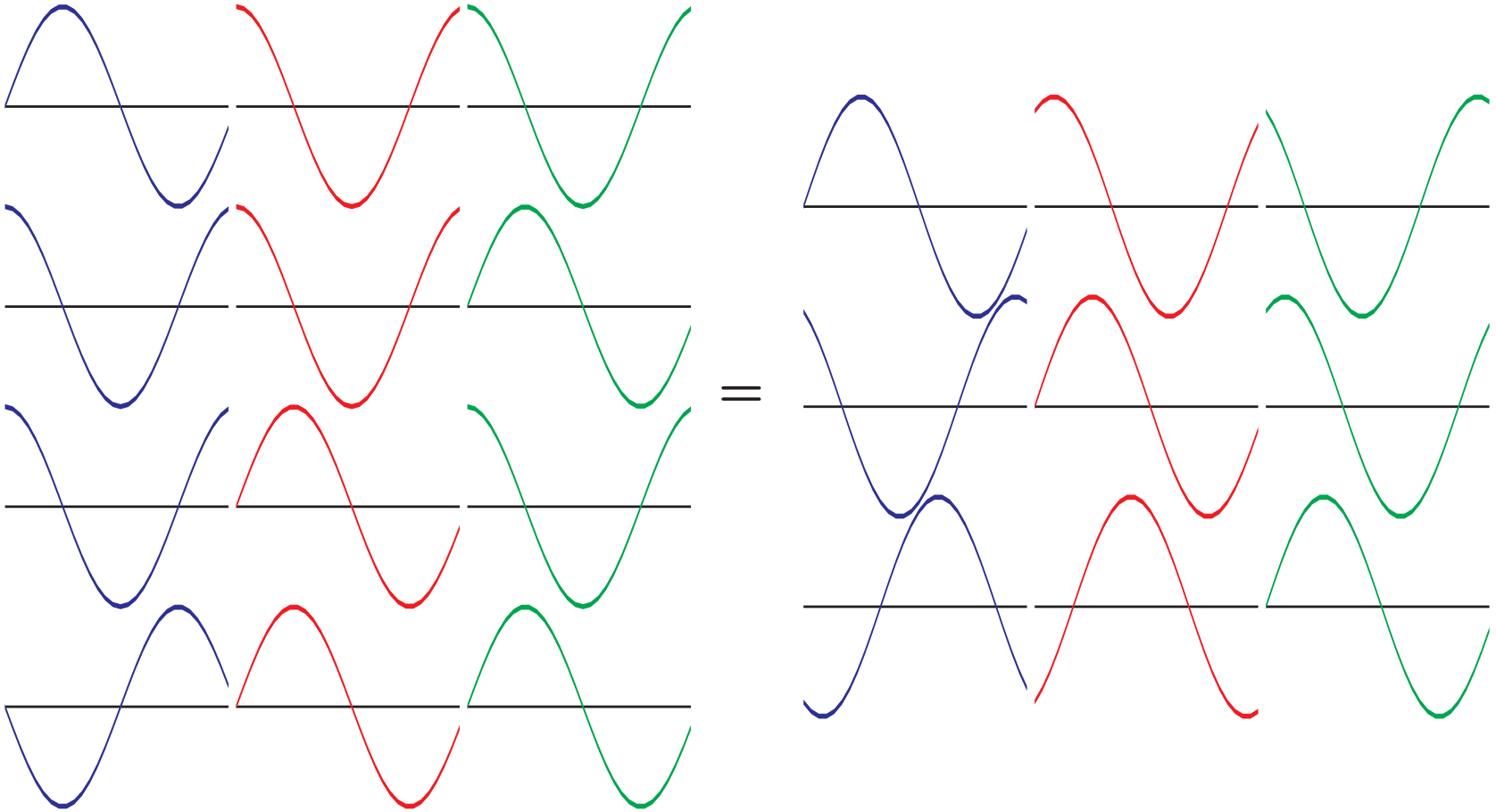
The standard trigonometric identities say $\sin(x_1 + \cdots + x_d)$ needs $r = 2^{d-1}$. For example, $d = 3 \Rightarrow r = 4$, and we have $\sin(\mathbf{x} + \mathbf{y} + \mathbf{z}) =$

$$\sin(\mathbf{x})\cos(\mathbf{y})\cos(\mathbf{z})$$
$$+ \cos(\mathbf{x})\cos(\mathbf{y})\sin(\mathbf{z})$$
$$+ \cos(\mathbf{x})\sin(\mathbf{y})\cos(\mathbf{z})$$
$$- \sin(\mathbf{x})\sin(\mathbf{y})\sin(\mathbf{z})$$

$=$



Would you have guessed?



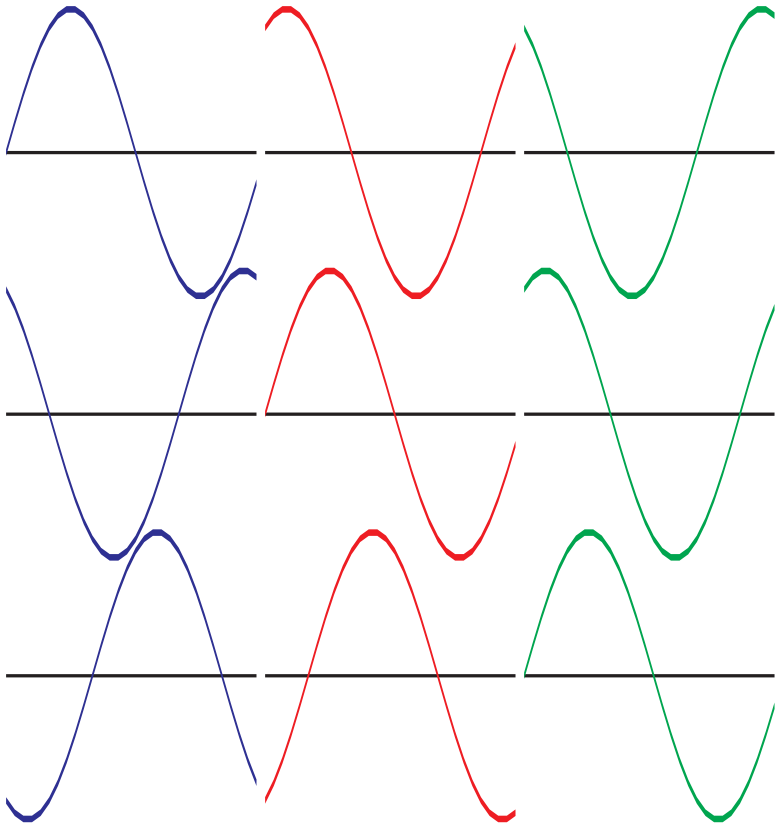
We discovered (numerically) a separation rank 3 representation: for arbitrary α, β, and γ, we have $\sin(\mathbf{x}+\mathbf{y}+\mathbf{z}) =$

$$\frac{\sin(\mathbf{x})\sin(\mathbf{y}+\beta-\alpha)\sin(\mathbf{z}+\gamma-\alpha)}{\sin(\beta-\alpha)\sin(\gamma-\alpha)}$$

$$+\frac{\sin(\mathbf{x}+\alpha-\beta)\sin(\mathbf{y})\sin(\mathbf{z}+\gamma-\beta)}{\sin(\alpha-\beta)\sin(\gamma-\beta)}$$

$$+\frac{\sin(\mathbf{x}+\alpha-\gamma)\sin(\mathbf{y}+\beta-\gamma)\sin(\mathbf{z})}{\sin(\alpha-\gamma)\sin(\beta-\gamma)}$$

$=$



Sine of the sum of several variables

Instead of $r = 2^{d-1}$ terms in dimension d,
we need only $r = d$ terms.

Theorem: As long as $s(\alpha_k - \alpha_j) \neq 0$ for all $j \neq k$, the functions

$$s(x) = a \exp(bx) x$$

and

$$s(x) = a \exp(bx) \sin(cx)$$

for any complex $a \neq 0$, b, and $c \neq 0$, satisfy

$$s\left(\sum_{j=1}^{d} x_j\right) = \sum_{j=1}^{d} s(x_j) \prod_{k=1, k\neq j}^{d} \frac{s(x_k + \alpha_k - \alpha_j)}{s(\alpha_k - \alpha_j)}$$

(Joint work with Lucas Monzón.)

Example: The Laplacian

The d-dimensional Laplacian apparently has separation rank d.
When $d = 4$ we have $\Delta =$

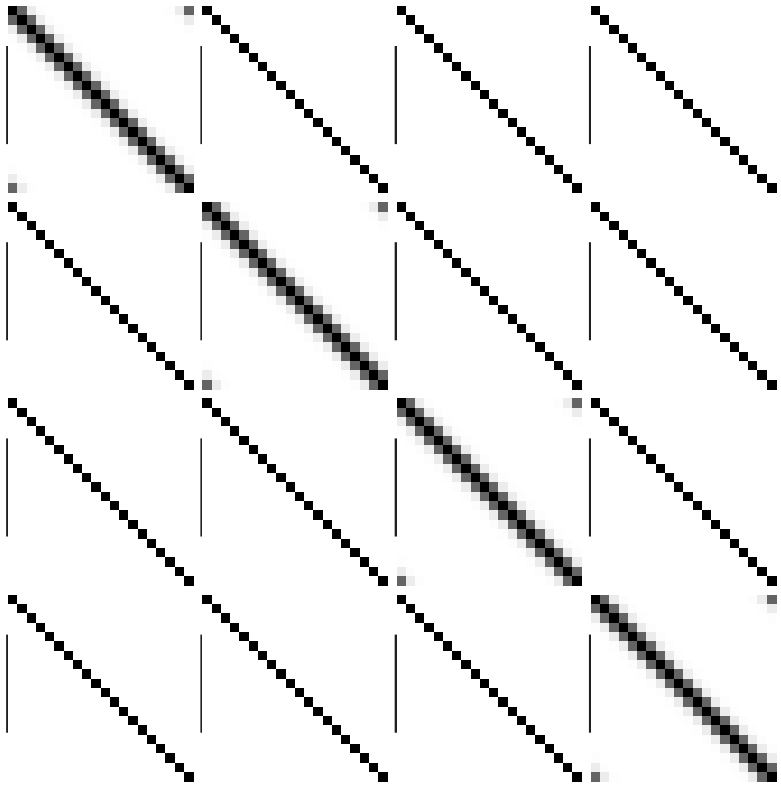
$$\frac{\partial^2}{\partial x_1^2} \otimes \mathcal{I}_2 \otimes \mathcal{I}_3 \otimes \mathcal{I}_4$$

$$+\mathcal{I}_1 \otimes \frac{\partial^2}{\partial x_2^2} \otimes \mathcal{I}_3 \otimes \mathcal{I}_4$$

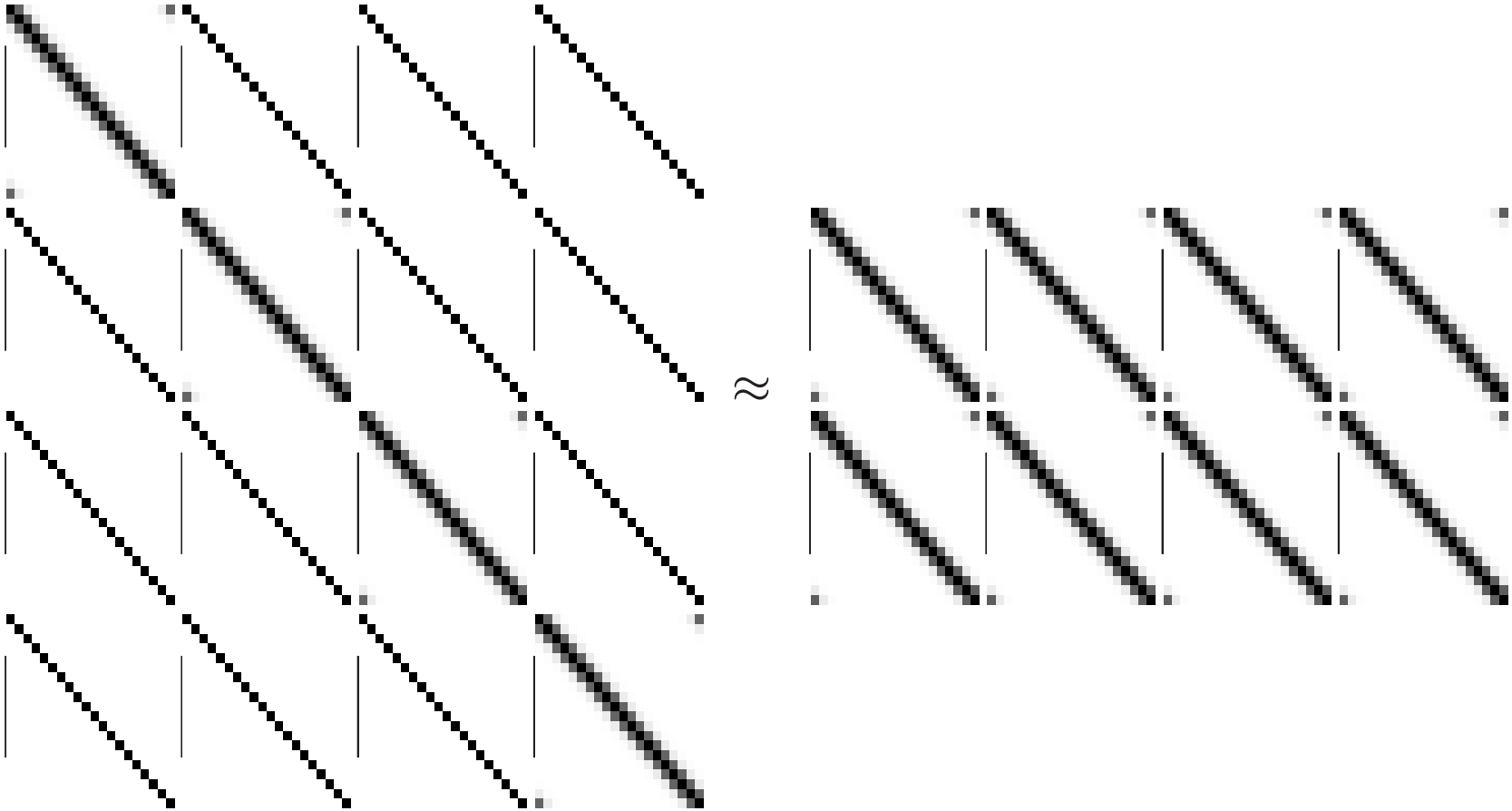
$$+\mathcal{I}_1 \otimes \mathcal{I}_2 \otimes \frac{\partial^2}{\partial x_3^2} \otimes \mathcal{I}_4$$

$$+\mathcal{I}_1 \otimes \mathcal{I}_2 \otimes \mathcal{I}_3 \otimes \frac{\partial^2}{\partial x_4^2}$$

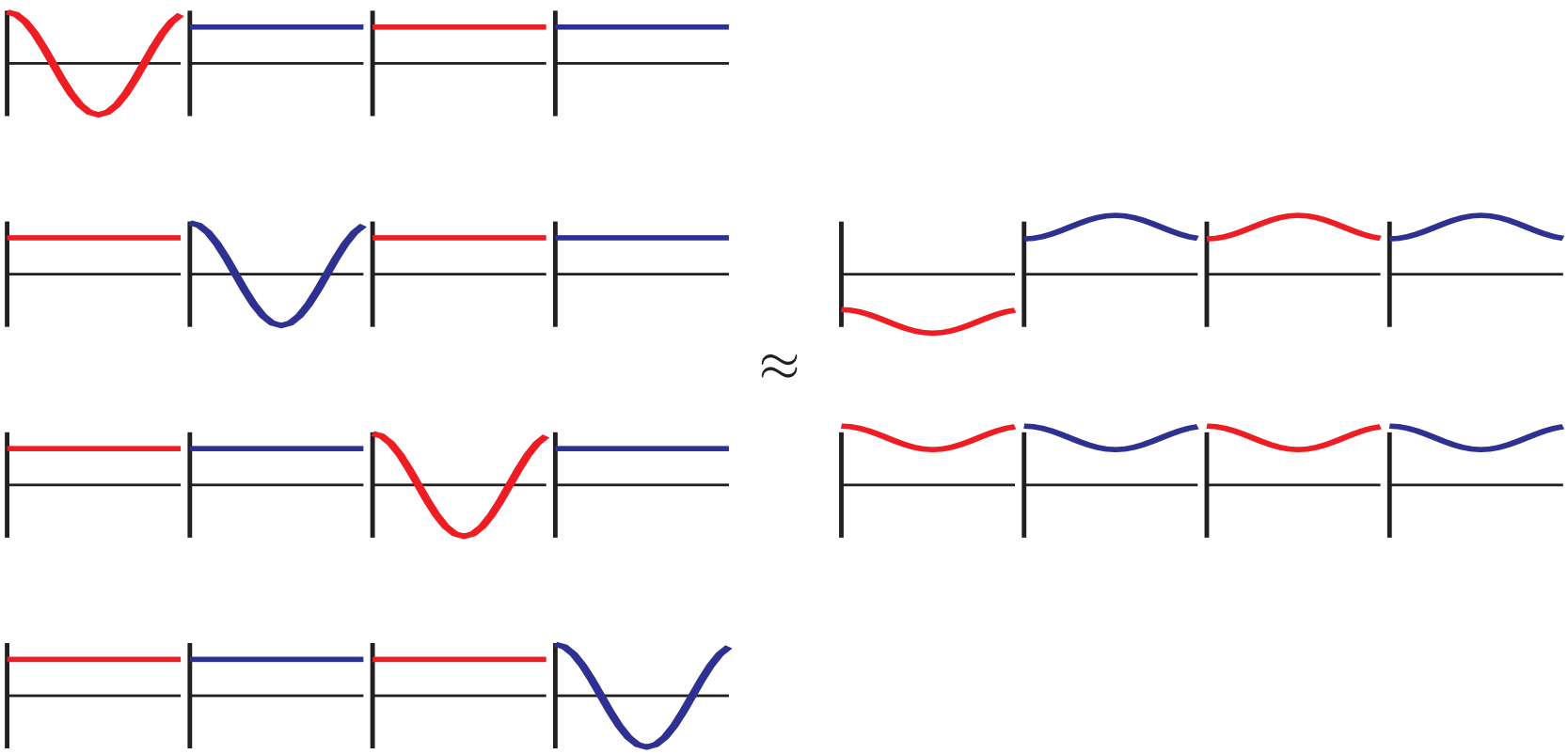
$=$



Would you have guessed?



Would you have guessed?



Example: Additive Model

$$f(\mathbf{x}) = \sum_{i=1}^{d} g_i(x_i) = \frac{d}{dt}\left[\prod_{i=1}^{d}(1 + t g_i(x_i))\right]_{t=0}$$
$$= \lim_{h\to 0} \frac{1}{2h}\left(\prod_{i=1}^{d}(1 + h g_i(x_i)) - \prod_{i=1}^{d}(1 - h g_i(x_i))\right),$$

so we can approximate a function that naively would have $r = d$ using only $r = 2$.

This formula provides a reduction of addition to multiplication; it is connected to exponentiation, since one could use $\exp(\pm h g_i(x_i))$ instead of $1 \pm h g_i(x_i)$.

We have just learned

- There is more going on here than meets the eye.
- The "obvious" (analytic) separated representation may be woefully inefficient.
- There may be entire families of separated representations.
- Some separated representations may have large separation values, leading to poor conditioning.

Example: Linear Model

For a linear model

$$f(\mathbf{x}) = \phi\left(\sum_{i=1}^{d} a_i x_i + b\right),$$

if we can write

$$\phi(t) \approx \sum_{l=1}^{r} \alpha_l \exp(\beta_l t)$$

then

$$f(\mathbf{x}) \approx \sum_{l=1}^{r} \alpha_l \exp(\beta_l(\sum a_i x_i + b)) = \sum_{l=1}^{r} \alpha_l \exp(\beta_l b) \prod_{i=1}^{d} \exp(\beta_l a_i x_i)\,.$$

A rotation of the coordinate axes would introduce constants in front of the x_i, but would not effect r.

Example: Gaussians

Gaussians are separable since

$$\exp(-c\|\mathbf{x}-\mathbf{z}\|^2) = \prod_{i=1}^{d} \exp(-c(x_i - z_i)^2)\,.$$

By expanding a radial function in Gaussians, one can obtain a separated representation for it.

Several operators, such as the inverse Laplacian, have radial kernels and thus can be represented using this technique.

Example: The Inverse Laplacian

Lemma: For any $0 < \delta < 1$, and $0 < \epsilon \leq \frac{1}{2}$ there exist positive numbers τ_l and σ_l such that for all $\delta \leq y \leq 1$

$$\left| \frac{1}{y^2} - \sum_{l=1}^{r} \sigma_l e^{-\tau_l y^2} \right| \leq \frac{\epsilon}{y^2} \,,$$

with

$$r = \log \epsilon^{-1} \left[c_0 + c_1 \log \epsilon^{-1} + c_2 \log \delta^{-1} \right] ,$$

where c_0, c_1, and c_2 are constants independent of ϵ and δ.

(due to Beylkin and Monzón)

Example: The Inverse Laplacian

We can obtain a separated representation for Δ^{-1}, valid on the annulus $(\sum_{i=1}^{d} \xi_i^2)^{1/2} \in [\delta, 1]$ in Fourier space, by substituting

$$y^2 = -\left(\frac{\partial^2}{\partial x_1^2} + \frac{\partial^2}{\partial x_2^2} + \cdots + \frac{\partial^2}{\partial x_d^2}\right).$$

We obtain

$$\Delta^{-1} \approx -\sum_{l=1}^{r} \sigma_l \exp(\tau_l \sum_{i=1}^{d} \frac{\partial^2}{\partial x_i^2}) = -\sum_{l=1}^{r} \sigma_l \bigotimes_{i=1}^{d} \exp(\tau_l \frac{\partial^2}{\partial x_i^2}).$$

(Rescale to get other domains.)

Classes of Functions with Low Separation Rank

Traditional approaches to characterizing a wide class of low-complexity functions use smoothness and decay of derivatives.

Temlyakov shows that for functions in the class W_2^k, which is characterized using partial derivatives of order k, there is a separated representation with separation rank r that has error

$$\epsilon = \mathcal{O}(r^{-kd/(d-1)}) .$$

However,

A careful analysis of the proof shows that

- the 'constant' in the $\mathcal{O}(\cdot)$ is at least $(d!)^{2k}$
- the inductive argument can only run if $r \geq d!$

Thus this result, while technically correct, does not provide a useful class of functions with low separation rank.

Challenge: Give a non-trivial characterization of functions with low separation rank.

Computational Paradigm, Strong Form

1. Start with operators/matrices and functions/vectors that can be represented within ϵ with low separation rank.

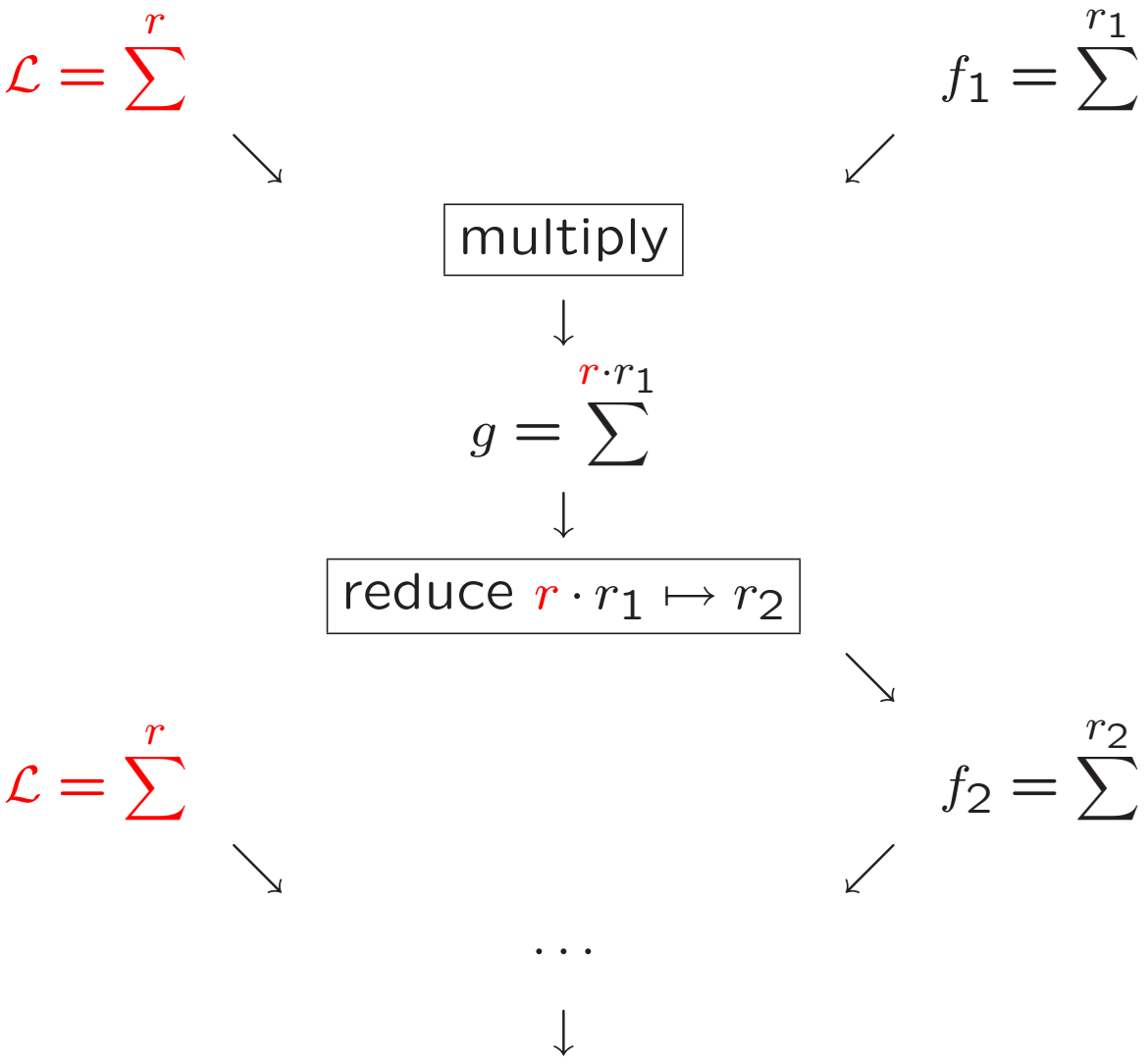
2. Do linear algebra operations with them, e.g.

$$g = \mathcal{L}f = \sum_{l=1}^{r} s_l \sum_{m=1}^{r_1} s_m^1 \bigotimes_{i=1}^{d} (\mathcal{L}_i^l f_i^m(x_i))$$

   The computational cost is $\mathcal{O}(d \cdot r \cdot r_1)$
   which is **linear** in d rather than exponential.

3. Adaptively re-minimize the separation rank of the output of each operation.

Example: Power Method



Compatible Algorithms

1. Power method ($f_{k+1} = \mathcal{L} f_k$) to compute an eigenvector.

2. Schulz iteration ($\mathcal{N}_{k+1} = 2\mathcal{N}_k - \mathcal{N}_k \mathcal{L} \mathcal{N}_k$) to construct $\mathcal{L}^{-1}$.

3. Sign iteration ($\mathcal{L}_{k+1} = (3\mathcal{L}_k - \mathcal{L}_k^3)/2$) to construct sign($\mathcal{L}$).

4. Scaling and squaring ($(\exp(\mathcal{L}/2^n))^{2^n}$) to construct exp($\mathcal{L}$).

5. . . .

Reducing the Separation Rank: Alternating Least Squares

We need to convert a sub-optimal separated representation $g = \sum_{m=1}^{r_g} s_m^g \bigotimes_{i=1}^{d} g_i^m$ to a (nearly) optimal one.

Iteratively refine an initial approximation $f = \sum_{l=1}^{r} s_l \bigotimes_{i=1}^{d} f_i^l$.

Loop through the directions $k = 1, \ldots, d$.

- Fix $\{f_i^l\}$ for $i \neq k$, and solve a **linear** least squares problem for new f_k^l and s_l.

If $\|f - g\|$ stabilizes but is too large, then increase r and repeat.
One full alternating least squares iteration costs

$$\mathcal{O}(d \cdot r(r^2 + r_g \cdot M)).$$

Refresher on the Usual Linear Least Squares Fitting

To find the coefficients $\{c_i\}$ to minimize

$$\left\| g - \sum_i c_i f_i \right\|^2 = \left\langle g - \sum_i c_i f_i, g - \sum_i c_i f_i \right\rangle ,$$

take the gradient with respect to $\{c_i\}$ and set it equal to 0 to obtain the normal equations

$$\mathbb{A}\mathbf{y} = \mathbf{b} ,$$

with

$$A(k,i) = \langle f_k, f_i \rangle \quad \text{and} \quad b(k) = \langle f_k, g \rangle .$$

As long as $\{f_i\}$ is linearly independent, the system has a unique solution $\mathbf{y}$, and one has $c_i = y(i)$.

Our $\mathbb{A}$ and $\mathbf{b}$

For $k = 1$, construct the matrix of integral operators $\mathbb{A}(x - x')$ with entries

$$A(l, l')(x - x') = \left\langle \delta(x - x_1) \prod_{i=2}^{d} f_i^l(x_i), \delta(x' - x_1) \prod_{i=2}^{d} f_i^{l'}(x_i) \right\rangle$$
$$= \delta(x - x') \prod_{i=2}^{d} \langle f_i^l, f_i^{l'} \rangle,$$

and the vector of functions $\mathbf{b}(x)$ with entries

$$b(l)(x) = \left\langle \delta(x - x_1) \prod_{i=2}^{d} f_i^l(x_i), \sum_{m=1}^{r^g} s_m^g \prod_{i=1}^{d} g_i^m(x_i) \right\rangle$$
$$= \sum_{m=1}^{r^g} s_m^g g_1^m(x) \prod_{i=2}^{d} \langle f_i^l, g_i^m \rangle.$$

The Normal Equations

The normal equations for direction 1 and coordinate x become

$$\int \mathbb{A}(x - x')\mathbf{y}(x')dx' = \mathbf{b}(x),$$

which reduces to

$$\mathbb{A}(0)\mathbf{y}(x) = \mathbf{b}(x).$$

This is just a linear system with multiple right-hand sides, indexed by x.

Solving for $\mathbf{y}(x)$ gives us $f_1^l = y(l)$, which we can then normalize and put the norms in s_l.

To use matrices/vectors instead of operators/functions, just replace the variable x with the index j.

Reducing the Separation Rank: Issues

- When $d > 2$ there is no satisfactory theoretical foundation or known way to compute an optimal representation.

- The Alternating Least Squares algorithm works well enough in practice.

- There are more sophisticated methods, based e.g. on Newton's method, but it is not clear if they are better.

Computational Paradigm, weak form

In some cases, the target for the least-squares error may not be explicitly available. Examples:

- We are trying to solve a linear system.
- There is a symmetry/antisymmetry constraint on the function.
- The operator involved does not have low separation rank.

To minimize least-squares error, we do not need to know our target explicitly, just to be able to take inner products.

A linear system in dimension d

Given a matrix

$$\mathbb{B} = \sum_{l=1}^{r_{\mathbb{B}}} s_l^{\mathbb{B}} \mathbb{B}_1^l \otimes \mathbb{B}_2^l \otimes \cdots \otimes \mathbb{B}_d^l,$$

and a right-hand-side vector

$$\mathbf{G} = \sum_{l=1}^{r_{\mathbf{G}}} s_l^{\mathbf{G}} \mathbf{G}_1^l \otimes \mathbf{G}_2^l \otimes \cdots \otimes \mathbf{G}_d^l,$$

we attempt to solve the linear system $\mathbb{B}\mathbf{F} = \mathbf{G}$ for $\mathbf{F}$ of the form

$$\mathbf{F} = \sum_{l=1}^{r_{\mathbf{F}}} s_l^{\mathbf{F}} \mathbf{F}_1^l \otimes \mathbf{F}_2^l \otimes \cdots \otimes \mathbf{F}_d^l,$$

by iteratively refining an approximate solution.

Alternating Least Squares

Recast $\mathbb{B}\mathbf{F} = \mathbf{G}$ as the minimization of the error $\|\mathbb{B}\mathbf{F} - \mathbf{G}\|$.

Start with random $\mathbf{F}$ with $r_{\mathbf{F}} = 1$.

- Loop while $\|\mathbb{B}\mathbf{F} - \mathbf{G}\| > \epsilon$:
  - Loop $k = 1, \ldots, d$:
    - Fix $\{\mathbf{F}_i^l\}_{i \neq k}$ and solve a linear least-squares problem for $\mathbf{F}_k^l$ (and $s_l^{\mathbf{F}}$) to minimize $\|\mathbb{B}\mathbf{F} - \mathbf{G}\|$.
  - If the relative decrease in $\|\mathbb{B}\mathbf{F} - \mathbf{G}\|$ is too small, then increase $r_{\mathbf{F}}$.

Form the matrix $\mathbb{A}$ with entries

$$A((\widehat{j},\widehat{l}),(j,l)) = \sum_{l_1}^{r_{\mathbb{B}}} \sum_{l_2}^{r_{\mathbb{B}}} s_{l_1}^{\mathbb{B}} s_{l_2}^{\mathbb{B}} \left(\sum_{j'}^{M} B_k^{l_1}(j',\widehat{j}) B_k^{l_2}(j',j) \right) \prod_{i \neq k} \langle \mathbb{B}_i^{l_1} \mathbf{F}_i^{l}, \mathbb{B}_i^{l_2} \mathbf{F}_i^{\widehat{l}} \rangle,$$

and the vector $\mathbf{b}$ with entries

$$b((\widehat{j},\widehat{l})) = \sum_{l_1}^{r_{\mathbb{B}}} s_{l_1}^{\mathbb{B}} \sum_{l_3}^{r_{\mathbf{G}}} s_{l_3}^{\mathbf{G}} \left(\sum_{j'}^{M} B_k^{l_1}(j',\widehat{j}) G_k^{l_3}(j') \right) \prod_{i \neq k} \langle \mathbf{G}_i^{l_3}, \mathbb{B}_i^{l_1} \mathbf{F}_i^{\widehat{l}} \rangle.$$

The normal equations for the direction k become

$$\mathbb{A}\mathbf{x} = \mathbf{b},$$

which we solve for $\mathbf{x} = x((j,l))$. Once we find $\mathbf{x}$, we compute $s_l^{\mathbf{F}} = \|x(\cdot,l)\|$ and set $F_k^l(j) = x(j,l)/s_l^{\mathbf{F}}$.

Assuming that M is the largest parameter, the dominant computational cost is $\mathcal{O}(dr_{\mathbf{F}}^3 M^3)$.

Preferred mode

Maintain error control throughout an algorithm by adjusting r as needed.

"Best effort" mode

When solving the linear system, we could not measure the error $\|\mathbf{F} - \mathbb{B}^{-1}\mathbf{G}\|$ but at least we could measure the residual $\|\mathbb{B}\mathbf{F} - \mathbf{G}\|$.

In some applications, we can minimize error through the least-squares fitting, but cannot measure the value of the error. We then fix r, minimize error, and hope.

Concluding Remarks

- You can do many things with sums of separable functions/operators.
- There are results on using this for regression in high dimensions.
- There are results on using this for the multiparticle Schrödinger equation.
- It would be great to have proofs that r is small.

Extra Slides

Controlling the Condition Number

The condition number is the ratio

$$\kappa = \frac{(\sum_{l=1}^{r} s_l^2)^{1/2}}{\|f\|}\,.$$

In order to maintain significant digits we need

$$\kappa\mu\|f\| \leq \epsilon\,,$$

where μ is the machine roundoff (e.g. 10^{-16}).

We can assure this by adding a penalty and minimizing $\|f-g\|^2 + 2\epsilon(\kappa\|f\|)^2$ instead. To minimize this all we have to do is add $2\epsilon\mathbb{I}$ to $\mathbb{A}$ in the alternating least-squares algorithm.

Ultimate Massive Parallel Super Computer

(Strömberg, ICM, 1998)

- Let the number of processors be the number of protons in the universe, $\approx 10^{80}$.
- Let the clock speed be the travel time for light to go one nuclear radius, $\approx 10^{19}$ Hz.
- Let the running time be the lifetime of the universe. (The age is $\approx 10^{17}$ s ; guess 10^4 ages.)

Then the total number of computations is

$$\approx 10^{120}$$

The Laplacian

Let

$$\mathcal{G}(t) = \bigotimes_{i=1}^{d} \left(\mathcal{I}_i + t \frac{\partial^2}{\partial x_i^2} \right)$$

and note that

$$\mathcal{G}'(t) = \Delta + \mathcal{O}(t)$$

so $\mathcal{G}'(0) = \Delta$. Using an appropriate finite difference formula of order r, we approximate

$$\mathcal{G}'(0) \approx \sum_{j=1}^{r} \alpha_j \mathcal{G}(t_j) \equiv \sum_{j=1}^{r} \alpha_j \bigotimes_{i=1}^{d} \left(\mathcal{I}_i + t_j \frac{\partial^2}{\partial x_i^2} \right),$$

thus providing a separation rank r approximation for Δ.

The Laplacian

Theorem Let $\mathcal{A}_i$ be a fixed, bounded operator $\mathcal{A}$ acting in the direction i, $\epsilon > 0$ be the error bound, and $0 < \mu \ll 1$ be the machine unit roundoff. Assuming $\mu d \|\mathcal{A}\| < \epsilon$, we can represent $\sum_i^d \mathcal{A}_i$ to within ϵ in the operator norm with separation rank

$$r = \mathcal{O}(\frac{\log(d\|\mathcal{A}\|/\epsilon)}{\log(1/\mu) - \log(d\|\mathcal{A}\|/\epsilon)}).$$

- We must first make the Laplacian bounded (by e.g. discretizing).

- The main behavior is $r \approx \log(d)$.