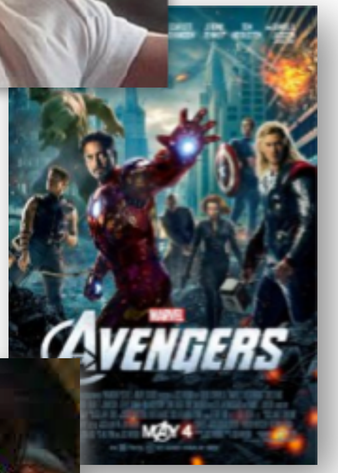
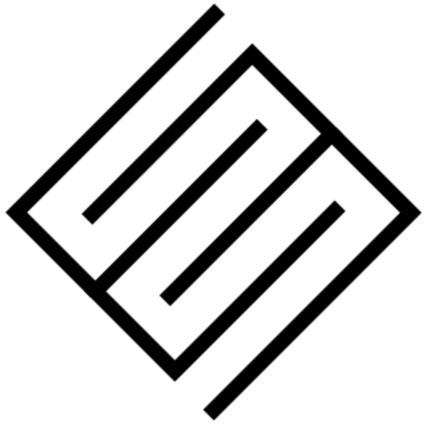
What the curse of dimensionality says about your face model

j.p.lewis

- Lead Researcher, SEED Electronic Arts
- adjunct faculty, Vic machine learning group





SEED // SEARCH FOR EXTRAORDINARY EXPERIENCES DIVISION

OUR FOCUS AREAS

Interactive characters

New types of experiences

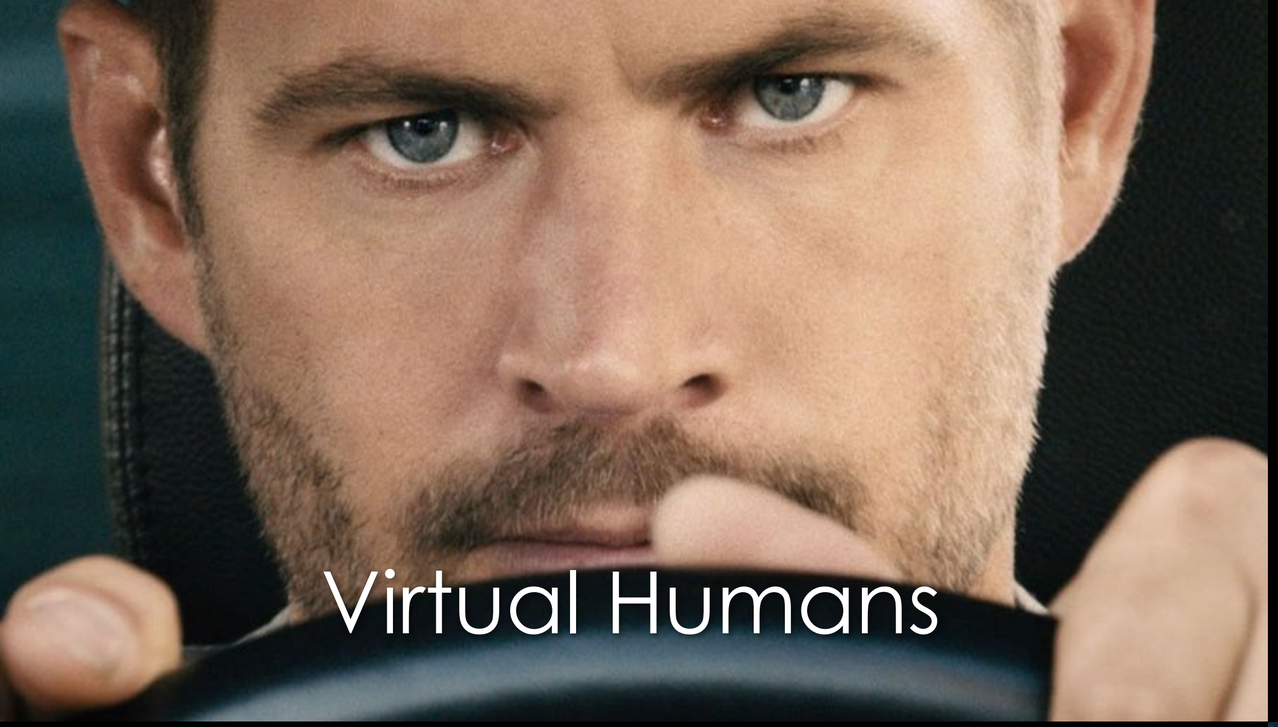
Make it easy to create games

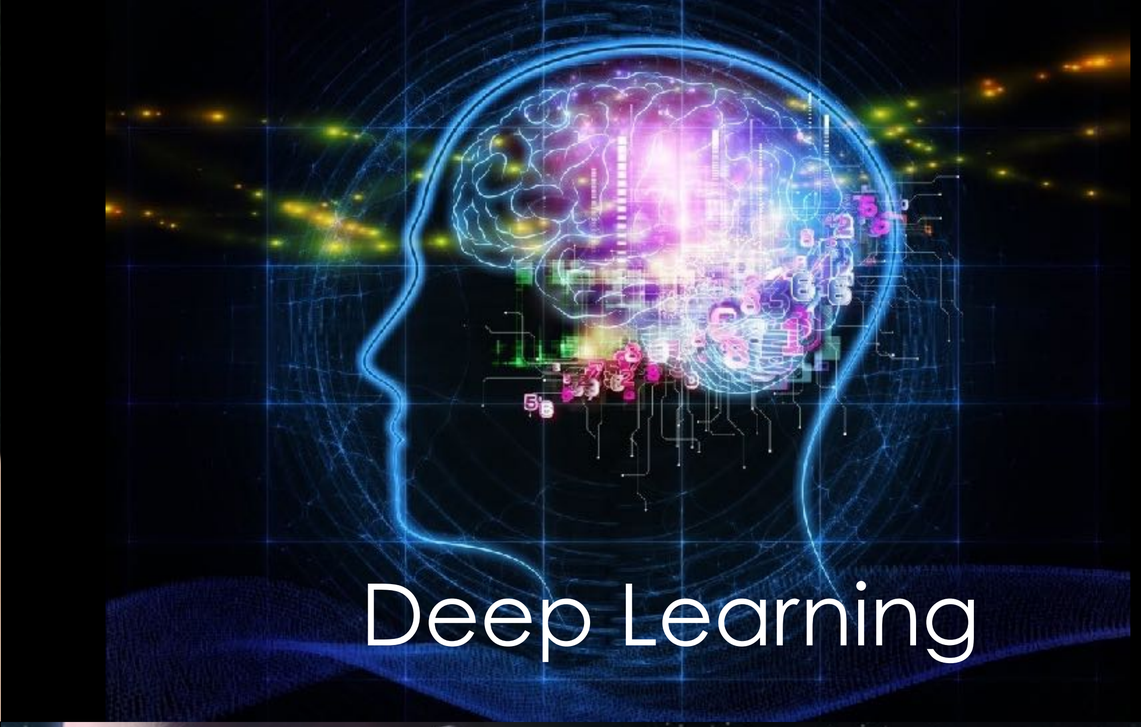
Foundational metaverse

TEAM

MIX OF CREATIVE & TECHNICAL

- Experience Designers
- Engineers
- AI Researchers
- Tech Artists
- Applied Mathematicians
- Computer Vision


Virtual Humans


Deep Learning

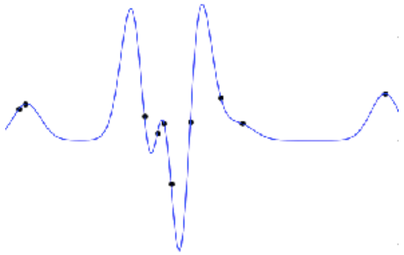
Prototypes

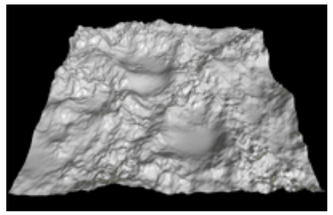
Future Worlds

dimension

dimension

- 1D function: one number in, one out. $R^1 \to R^1$.
  Graph lives in 2D
- 2D function: two numbers in, one out. $R^2 \to R^1$.
  Lives in 3D

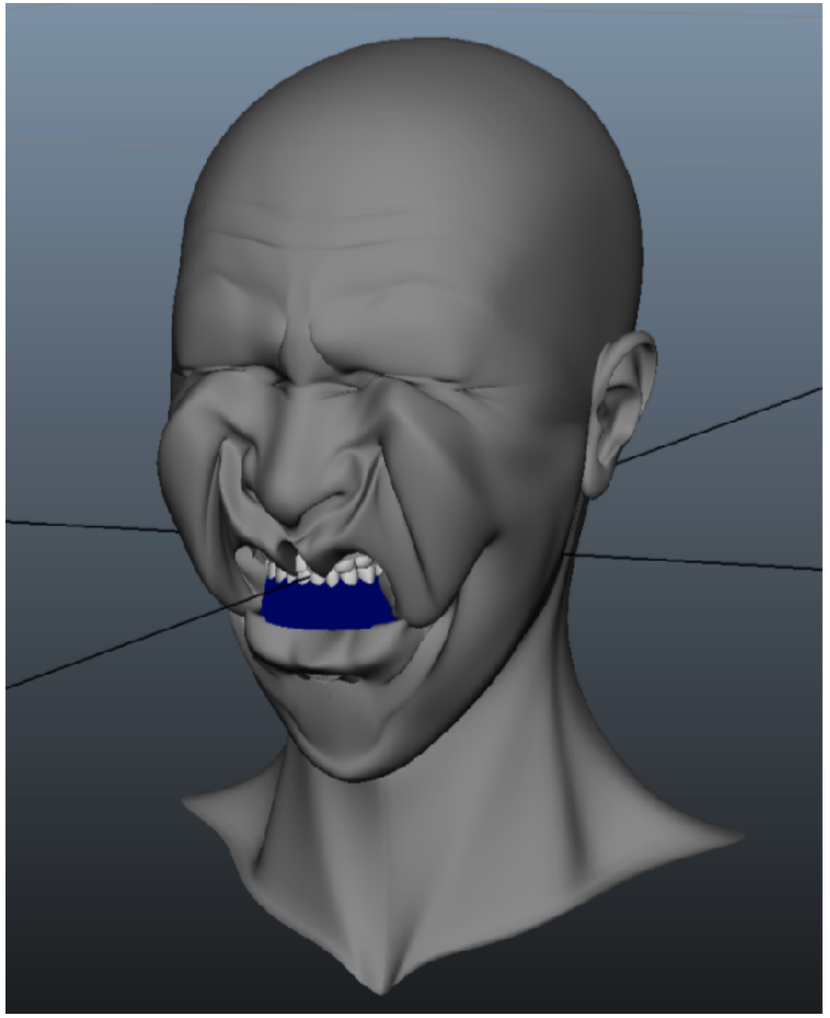




dimension

- megapixel image = **point** in million dimensional space
- blendshape head model with 1000 verts, 100 targets

dimension



manifold assumption

- “manifold = surface” (generalized to arbitrary dimensions)

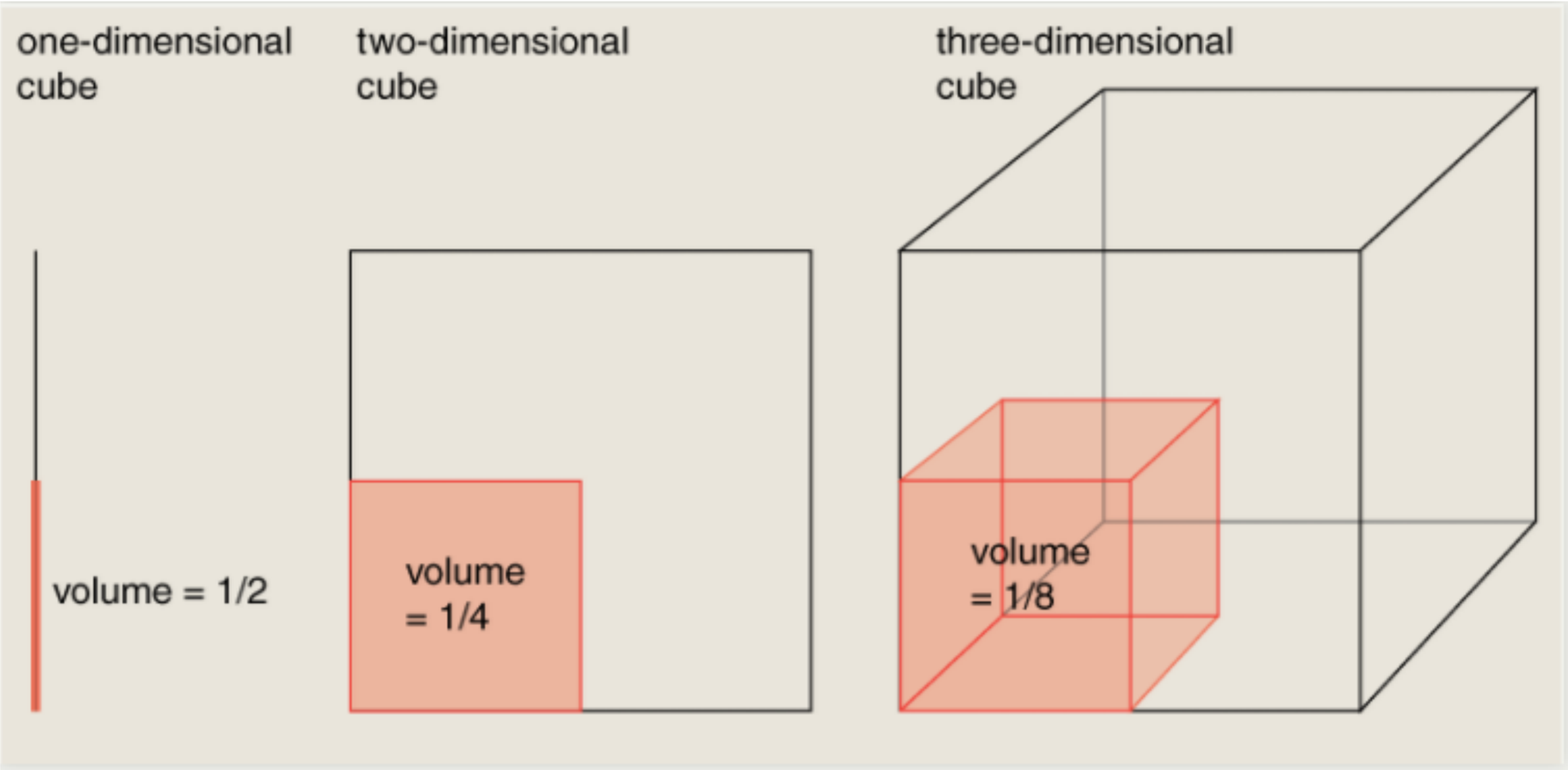
Euclidean distance in high dimensions

$$\begin{aligned}
\text{2D distance} &\qquad \sqrt{dx^2 + dy^2} \\
\text{3D distance} &\qquad \sqrt{dx^2 + dy^2 + dz^2} \\
\text{4D distance} &\qquad \sqrt{d_1^2 + d_2^2 + d_3^2 + d_4^2} \\
\text{ND distance} &\qquad \sqrt{d_1^2 + d_2^2 + d_3^2 + d_4^2 + \cdots}
\end{aligned}$$

curse of dimensionality:
though algorithms may generalize, intuitions do not

curse of dimensionality 1.

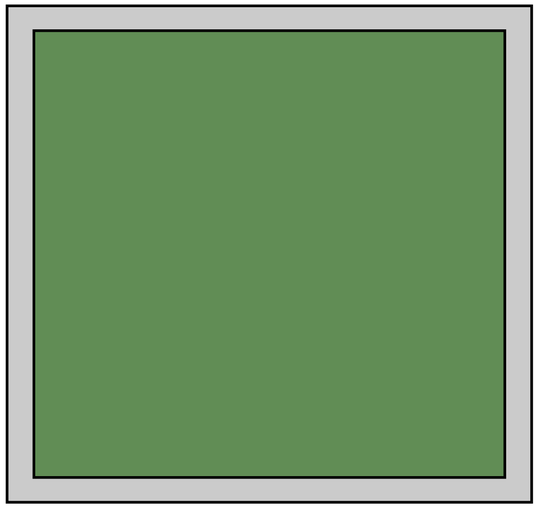
- machine learning algorithm requires data exponential in number of dimensions

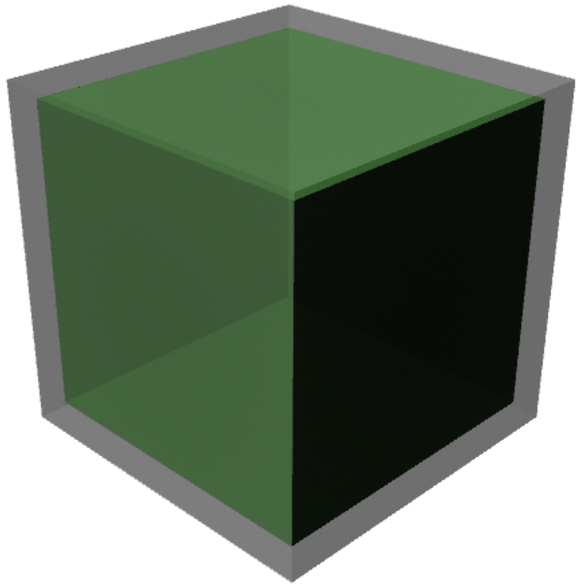


Brian Hayes

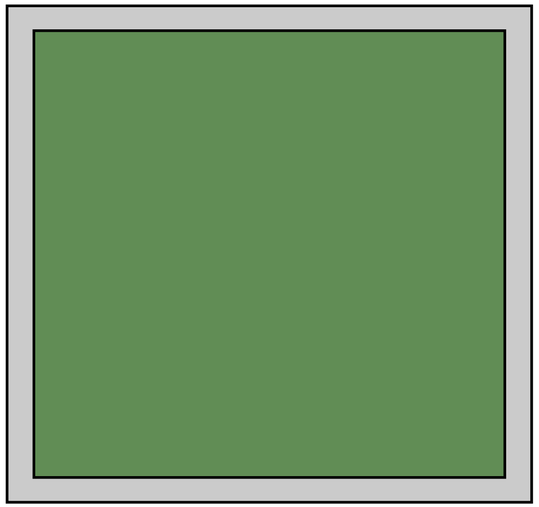
curse of dimensionality 2:
everything is near the surface. there is no volume.

curse of dimensionality 2:
everything is near the surface. there is no volume.





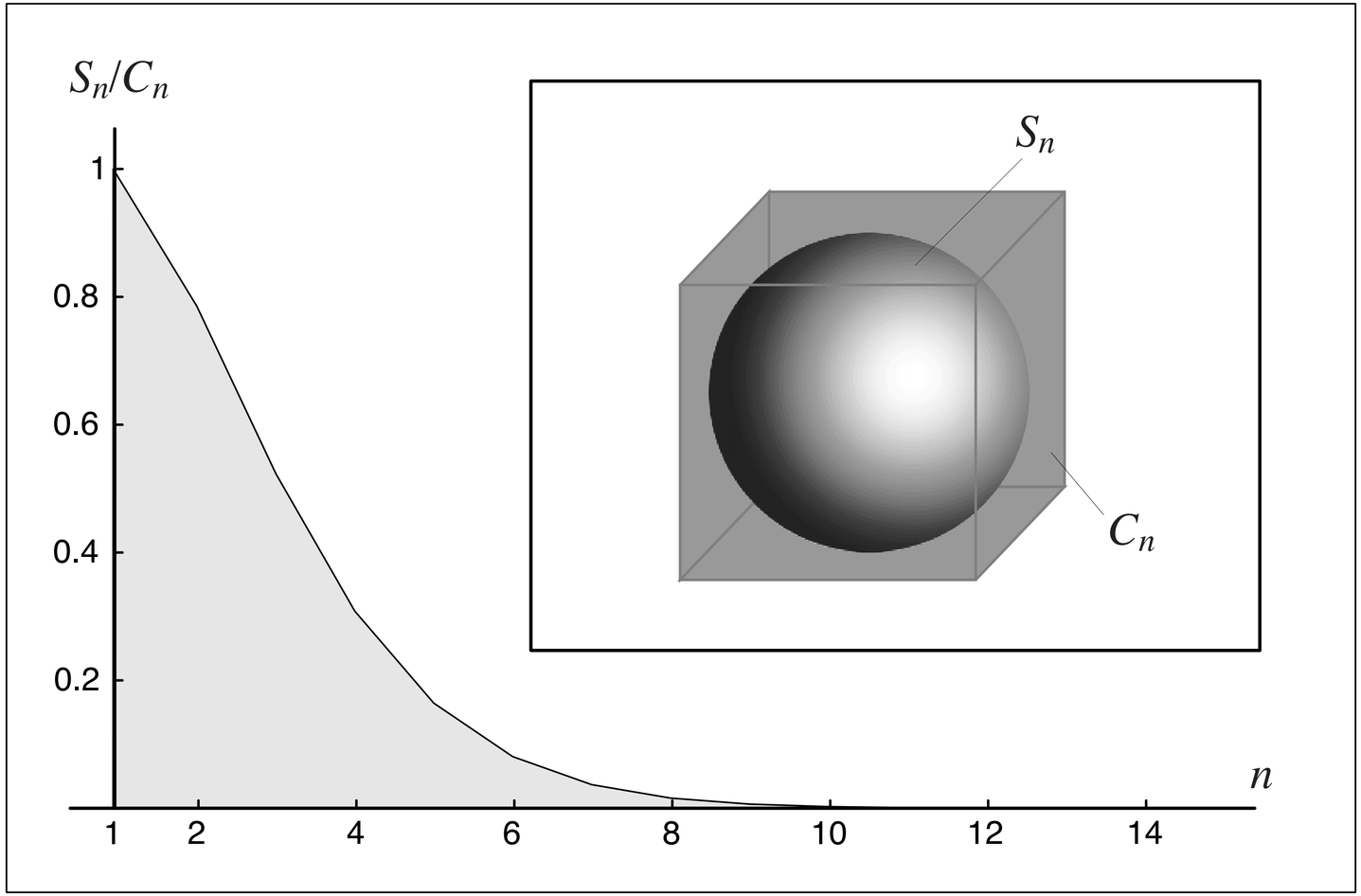
curse of dimensionality 2:
everything is near the surface. there is no volume.



volume of unit (hyper)cube: $1^D = 1$

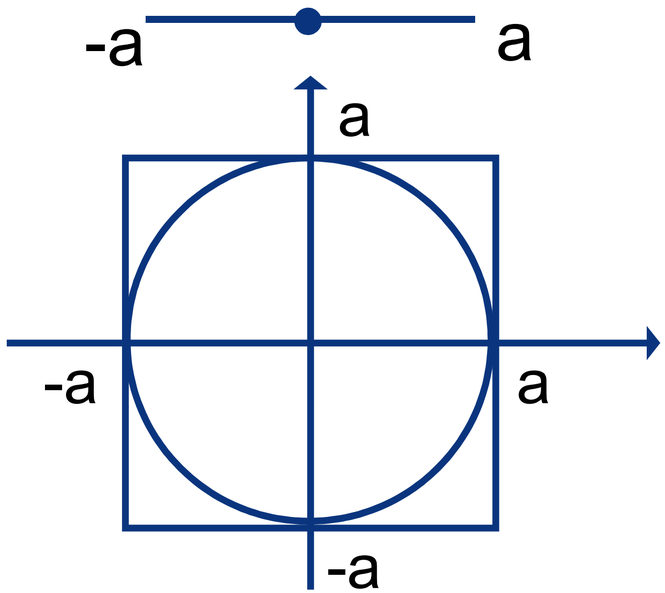
volume of .95 (hyper)cube: $.95^D \to 0$

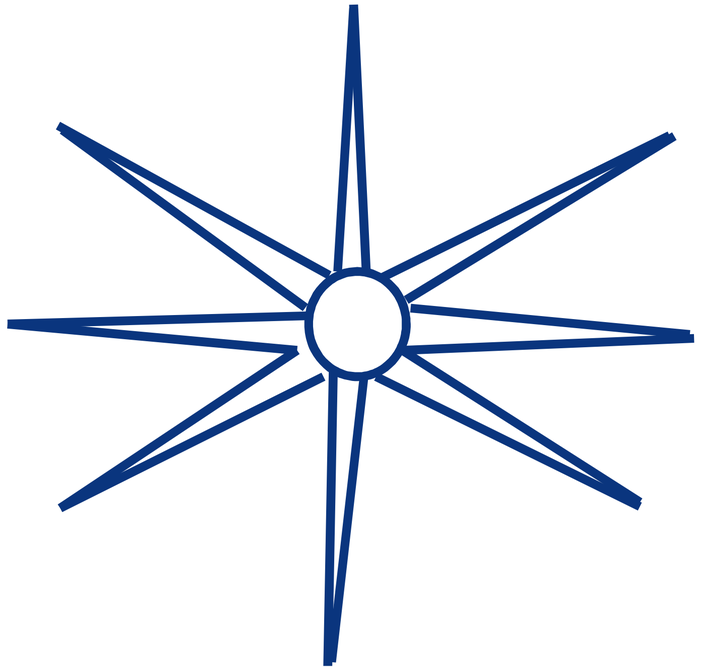
sphere vs cube volume



Judd

curse of dimensionality 2: everything is near the surface

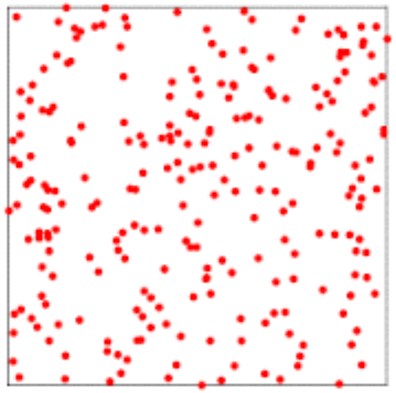


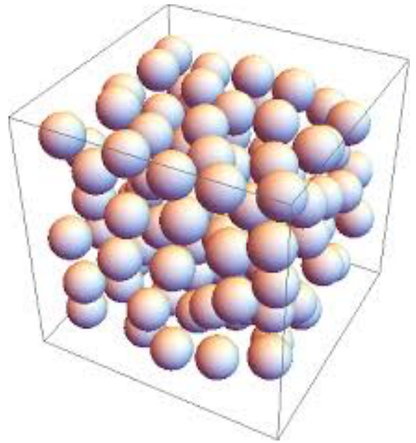


Vasconcelos

curse of dimensionality 3: everything is far away

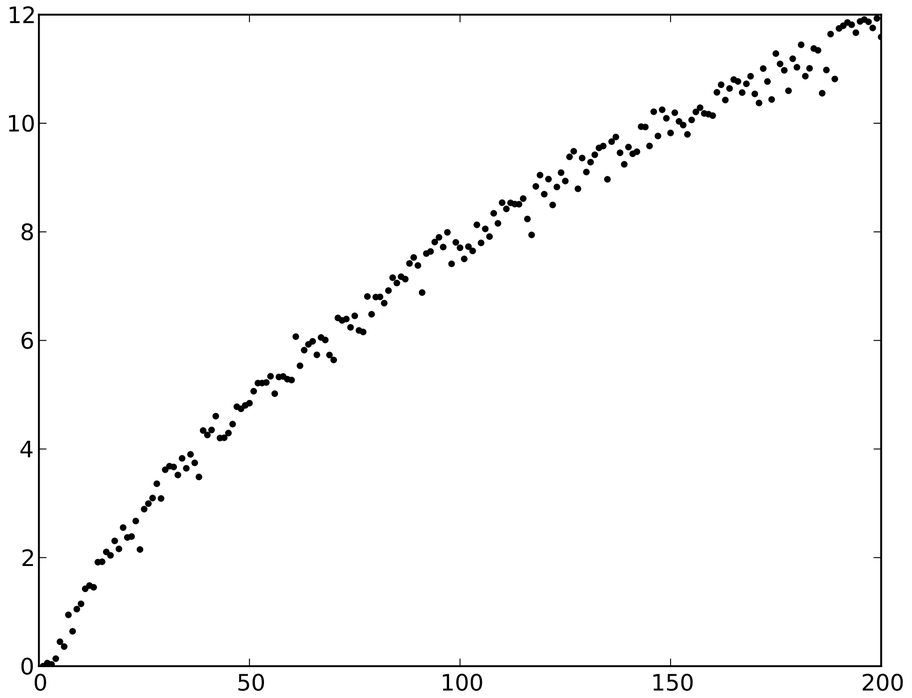
probe space with pseudo-randomly chosen points





in high dimensions everything is far away

- The closest distance (to a particular point) among 1000 unit-variance multidimensional Gaussian random variables



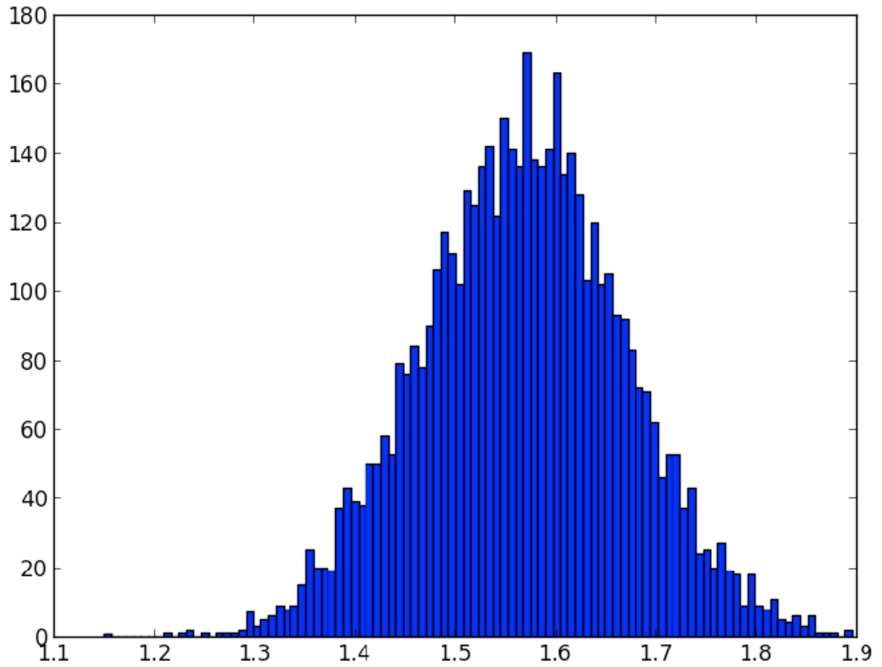
curse of dimensionality 3b: random points are almost the same distance away

- c.f. law of large numbers
- practical problem: find closest point using spatial data structure

curse of dimensionality 4: all directions are nearly perpendicular

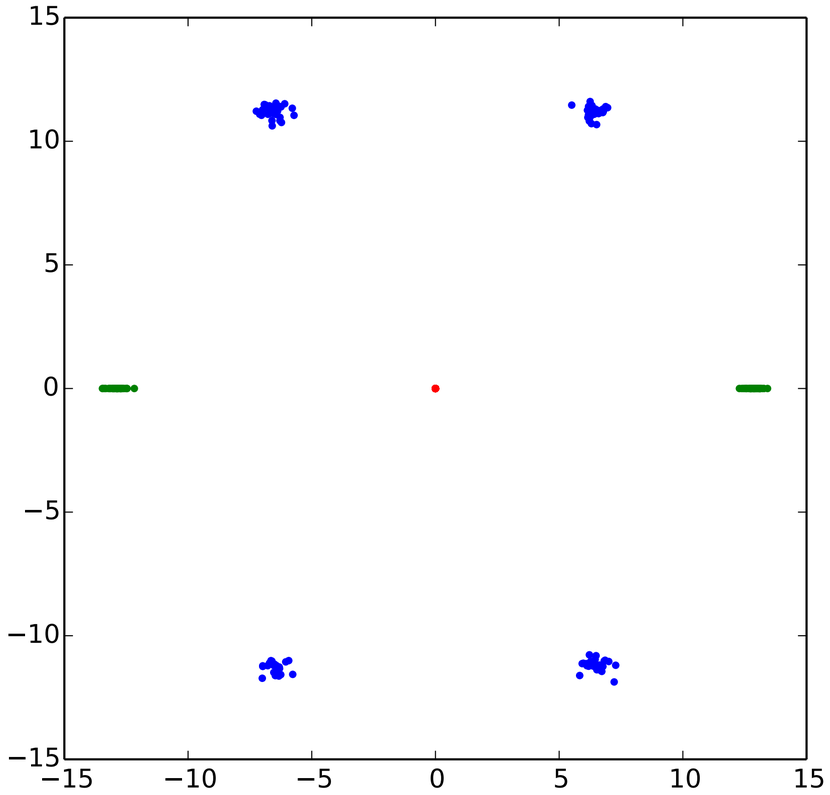
```
import math
import numpy as n_
import numpy.random as r_
import pylab as p_


d = 1000 # dimension
N = 100  # number of points

G = r_.multivariate_normal( n_.zeros(N), cov=n_.eye(N), size=(d))
print G.shape                   # (N,d)

for i in range(N):
    v = G[i,:]
    v = v / n_.linalg.norm(v)
    G[i,:] = v

Npairs = N*(N-1)/2
angles = n_.zeros((Npairs,1))
iangle = 0
for ir in range(N):
    v1 = G[ir,:]
    v1 = v1.T
    for ic in range(ir+1,N):
        v2 = G[ic,:]
        a = n_.dot(v1,v2)
        a = math.acos(a)
        angles[iangle] = a
        print a
        iangle = iangle + 1

#p_.hist(angles,100,histtype = 'stepfilled')
p_.hist(angles,100)
p_.savefig('hist_of_angles_%d_dimensions.pdf' % N)
```

random high-dimensional vectors tend to be orthogonal

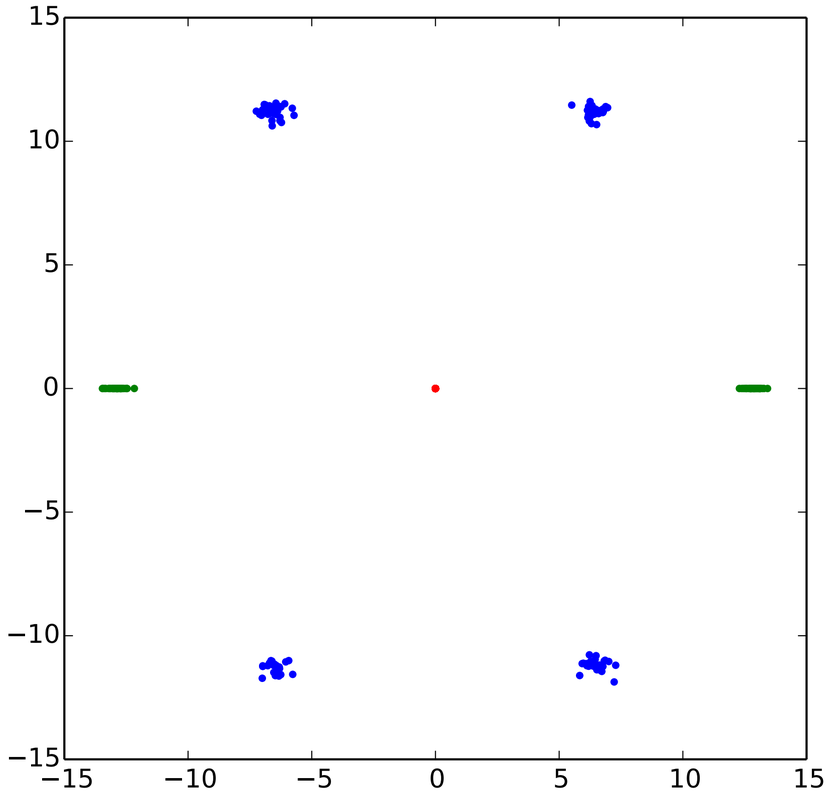


!

combine 3,4:

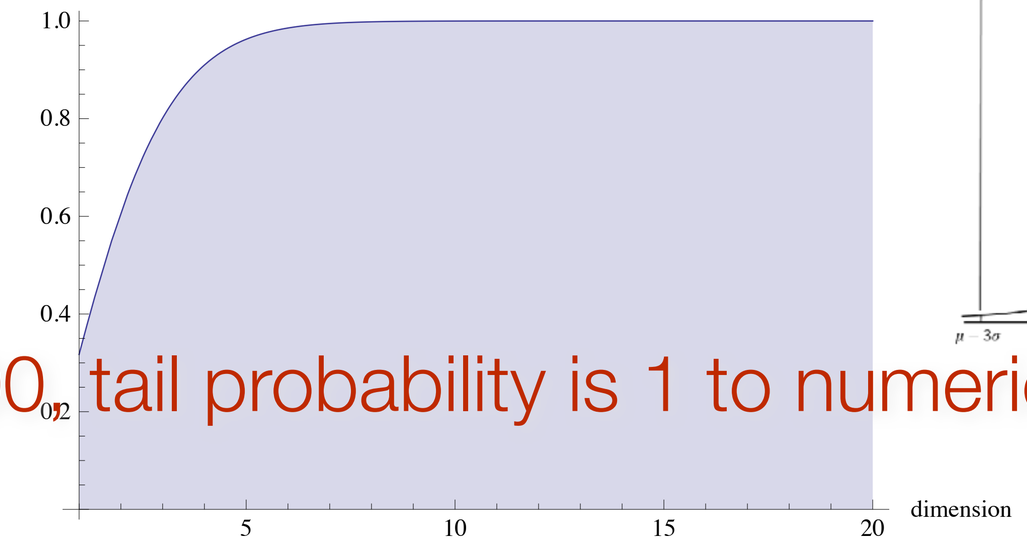


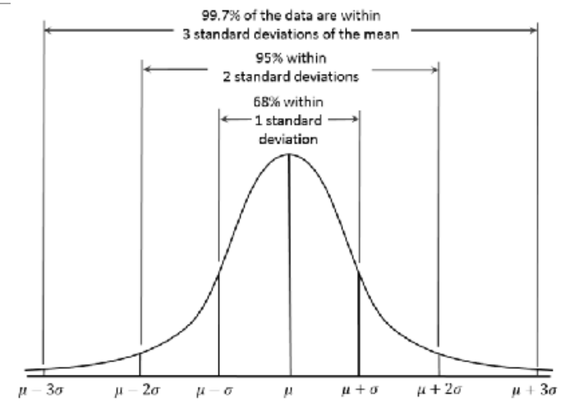
combine 3,4:
randomly chosen points lie on a randomly rotated+translated lattice



the high-dimensional Gaussian:
is also weird

in high dimensions the Gaussian is a heavy-tailed distribution!





with d=100, tail probability is 1 to numeric precision

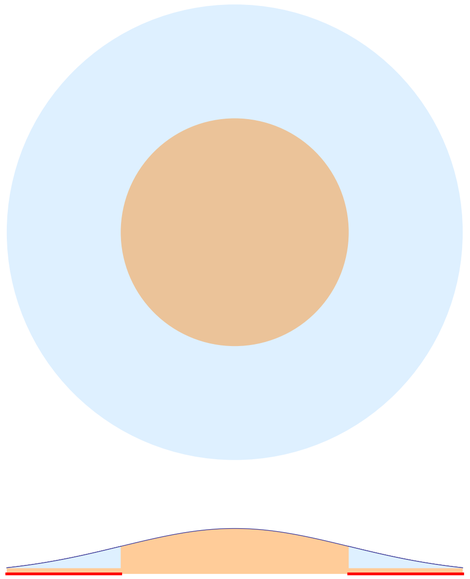
$$G(r) = \frac{1}{\sqrt{(2\pi)^d}} \exp^{-r^2/2}$$

$$\int_0^r S_{d-1}(r)G(r) = \frac{2\pi^{d/2}}{\Gamma(d/2)} \int_0^r r^{d-1}G(r)dr$$

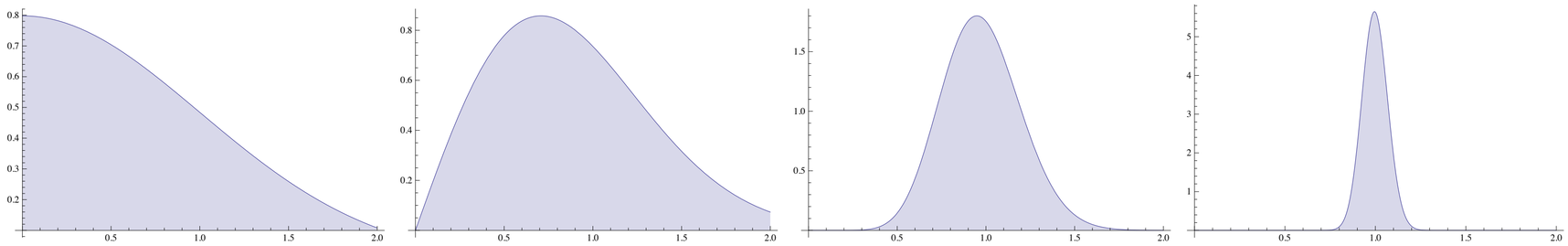
all the volume is concentrated in the tail

- Obvious for hypercube case:

$$(1-\epsilon)^d \to 0$$



probability mass concentrated in a hyperellipsoidal shell



- radially integrated unit variance normal in 1,2,10,100 dimensions

- Dirac delta: width is zero, integral is 1

- High-dimensional Gaussian near zero:
  width is non-zero, integral is (nearly) zero

So things are weird. Does it matter?

faces as points in 100 dimensions

- 10000 vertices => point in 30k dimensional space
- 100 blendshape targets => 100-dimensional subspace of 30k dimensional space
- estimates of between 40-100 dimensions for a linear face model [MS07, PS00, MXB06]
- => for discussion: 100-dimensional space with assumed Gaussian distribution

linear Gaussian models

- Graphics
  - Blanz & Vetter, morphable model (2000+ citations)
    many others
- Vision
  - Cootes & Taylor, Active Appearance Models
    many others

linear Gaussian models

- Psychology:

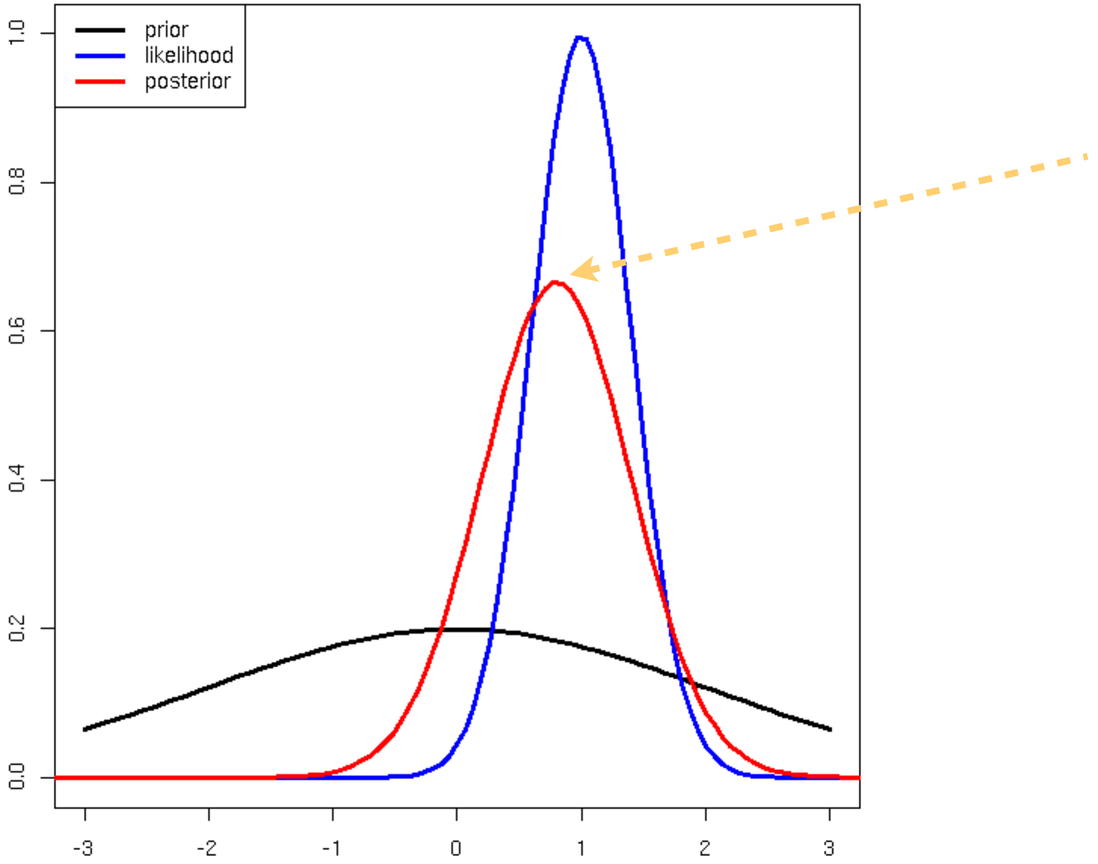
  “even for a face space of high dimensionality, the assumption of a multivariate normal distribution means that... There will be many typical faces that will be located relatively close to the center”.

  - Valentine T.: “Face-Space Models of Face Recognition” in Computational, Geometric, and Process Perspectives on Facial Cognition: Contexts and Challenges.

T. Valentine, A unified account of the effects of distinctiveness, inversion, and race in face recognition (800 citations)

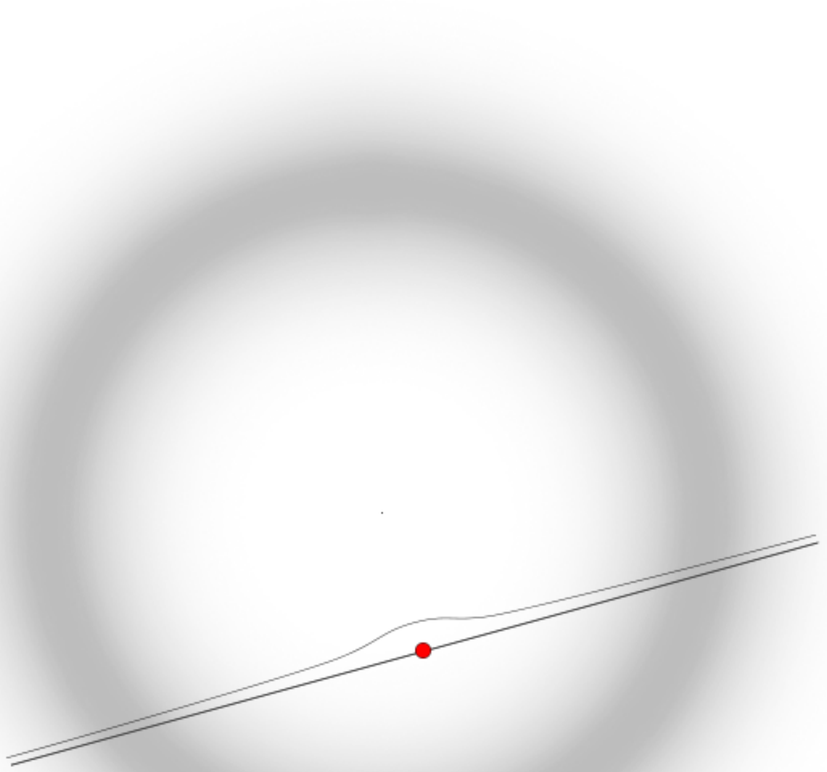
maximum a posteriori (MAP)

- given the data, find the model that fits the data
- allow a prior probability P(M)
- P(**M**|D) = P(D|M) P(M) / P(D)
- Gaussian assumption => pick the peak (mode) of the posterior
- Prior especially important for inverse problems


prior
likelihood
posterior
1.0
0.8
0.6
0.4
0.2
0.0
-3
-2
-1
0
1
2
3

MAP versus the curse: no volume in the center

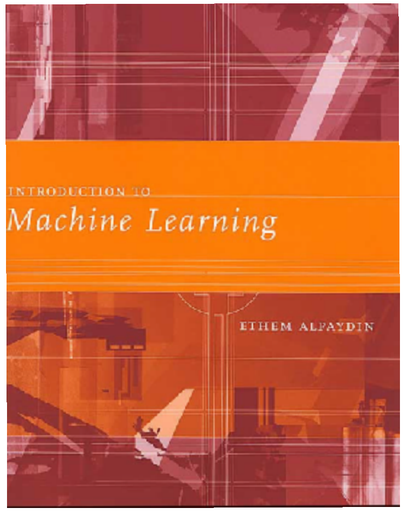
This plot is wrong



“even for a face space of high dimensionality, the assumption of a multivariate normal distribution means that... **There will be many typical faces that will be located relatively close to the center**”.

- Valentine T.: “Face-Space Models of Face Recognition” in Computational, Geometric, and Process Perspectives on Facial Cognition: Contexts and Challenges.

What about MAP?



Evaluating the integrals may be quite difficult, except in cases where the posterior has a nice form. When the full integration is not feasible, we reduce it to a single point. If we can assume that $p(\theta|\mathcal{X})$ has a narrow peak around its mode, then using the *maximum a posteriori* (MAP) *estimate* will make the calculation easier:

MAXIMUM A POSTERIORI ESTIMATE

$$\theta_{MAP} = \arg\max_{\theta} p(\theta|\mathcal{X}) \tag{4.14}$$

thus replacing a whole density with a single point, getting rid of the integral and using as

$$p(x|\mathcal{X}) = p(x|\theta_{MAP})$$
$$y_{MAP} = g(x|\theta_{MAP})$$

If we have no prior reason to favor some values of θ, then the prior density is flat and the posterior will have the same form as the likelihood, $p(\mathcal{X}|\theta)$, and the MAP estimate will be equivalent to the maximum likelihood estimate (section 4.2) where we have

$$\theta_{ML} = \arg\max_{\theta} p(\mathcal{X}|\theta) \tag{4.15}$$

BAYES' ESTIMATOR

Another possibility is the *Bayes' estimator*, which is defined as the expected value of the posterior density

What about MAP?

- probability density maxima often have very little associated probability mass even though the value of the probability density there may be immense, because they have so little associated volume...

  the locations of probability density maxima in many dimensions are generally misleading and irrelevant. Probability densities should only be maximized if there is good reason to believe that the location of the maximum conveys useful information about the whole distribution."

  - D. Mackay, 1996

What about MAP?

- probability density maxima often have very little associated probability mass even though the value of the probability density there may be immense, because they have so little associated volume...

  the locations of probability density maxima in many dimensions are generally misleading and irrelevant. Probability densities should only be maximized if there is good reason to believe that the location of the maximum conveys useful information about the whole distribution."

  - D. Mackay, Hyperparameters: Optimize, or integrate out? In Maximum entropy and Bayesian methods, 1996

Does it matter?

Does it matter?

Does it matter?

Does it matter?

Probable and Improbable Faces

J.P. Lewis, Zhenyao Mo, Ken Anjyo, Taehyun Rhee

Mathematics of Expressive Image Synthesis (MEIS) 2013

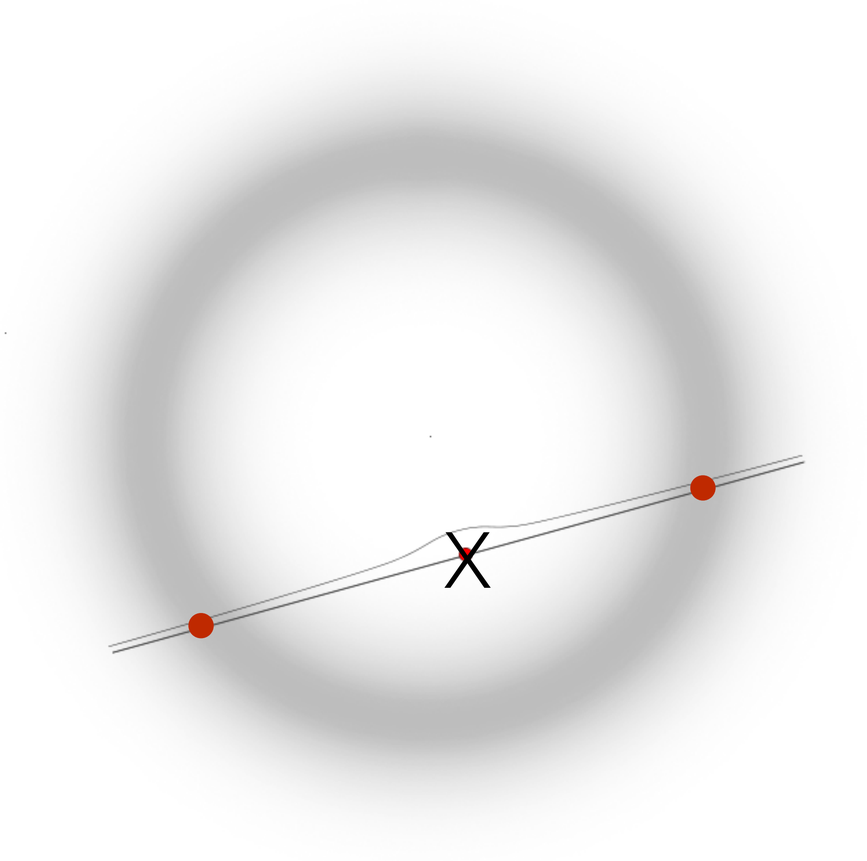
The End

---

https://www.ea.com/seed

http://scribblethink.org
noisebrain@gmail.com

Where next?



Part II: randomized linear algebra

The Johnson-Lindenstrauss lemma

- n points can be embedded in a space of dimensionality proportional to log n (regardless of their dimensionality!) while preserving inter-point distances

Theorem 1 (Johnson-Lindenstrauss 1984) *Let $x_1, \ldots, x_n \in \mathbb{R}^d$ be arbitrary. Pick any $\epsilon = (0, 1)$. Then for some $t = O(\log(n)/\epsilon^2)$ there exist points $y_1, \ldots, y_n \in \mathbb{R}^t$ such that*

$$\begin{array}{rcccll} (1-\epsilon)\|x_j\| & \leq & \|y_j\| & \leq & (1+\epsilon)\|x_j\| & \forall j \\ (1-\epsilon)\|x_j - x_{j'}\| & \leq & \|y_j - y_{j'}\| & \leq & (1+\epsilon)\|x_j - x_{j'}\| & \forall j, j'. \end{array} \quad (1)$$

- does not depend on the data
- does not depend on the dimension
- can be achieved by multiplying the original points through a **random matrix** (Gaussian entries)

The Johnson-Lindenstrauss lemma

- n points can be embedded in a space of dimensionality proportional to log n (regardless of their dimensionality!) while preserving inter-point distances

- my opinion, not directly useful for us

- but inspired approach of random “projection”

intuition:
randomly chosen points will not fall exactly in a null space

fast SVD alternative

If matrix $\mathbf{A}$ is low rank, then it will be true that

$$\mathbf{A} \approx \mathbf{Q}\mathbf{Q}^T\mathbf{A} = \mathbf{Q}\mathbf{B} \qquad \text{with } \mathbf{B} = \mathbf{Q}^T\mathbf{A}$$

with $\mathbf{Q}$ a tall matrix. We want to find $\mathbf{Q}$ for less than the $O(N^3)$ cost that an SVD would require.

1. Pick a tall random matrix $\mathbf{R}$, and form the product $\mathbf{AR}$. This has the interpretation of pushing random vectors through the matrix $\mathbf{A}$.
2. Orthonormalized the result using QR (Gram-Schmidt):

$$\mathbf{Q}, r = \text{qr}(\mathbf{AR})$$

   Here $\mathbf{Q}$ is a tall matrix (r is the upper triangular matrix, not needed).
3. From the original statement of low-rank approximation:

$$\mathbf{A} \approx \mathbf{Q}\mathbf{Q}^T\mathbf{A} = \mathbf{QB} \qquad \text{with } \mathbf{B} = \mathbf{Q}^T\mathbf{A}$$

   Here $\mathbf{B}$ is a wide matrix.
4. To obtain an SVD-like decomposition, obtain the SVD of $\mathbf{B}$ and regroup,

$$\mathbf{A} \approx \mathbf{Q}\mathbf{Q}^T\mathbf{A} = \mathbf{QB} = \mathbf{Q}(\mathbf{USV}^T) = (\mathbf{QU})\mathbf{SV}^T$$

fast eigen decomposition

For a symmetric matrix, $\mathbf{Q}$ forms a basis for both row and column spaces, so

$$\begin{aligned}\mathbf{A} &\approx \mathbf{Q}\mathbf{Q}^T\mathbf{A}\mathbf{Q}\mathbf{Q}^T \\ &= \mathbf{Q}(\mathbf{Q}^T\mathbf{A}\mathbf{Q})\mathbf{Q}^T \\ &= \mathbf{Q}(\mathbf{V}\mathbf{\Lambda}\mathbf{V}^T)\mathbf{Q}^T \\ &= (\mathbf{Q}\mathbf{V})\mathbf{\Lambda}(\mathbf{V}^T\mathbf{Q}^T)\end{aligned}$$

Kaczmarz: “POCS for Ax=b”

$y =$ current solution

$x =$ projecton on plane $a'x = b$

$$\min (y-x)'(y-x) + \lambda(a'x - b)$$

$$\min y'y - 2y'x + x'x + \lambda(a'x - b)$$

$$2x - 2y + \lambda a = 0 \quad \text{and} \quad ax = b$$

$$x - y + \lambda a = 0 \quad \text{and} \quad ax = b$$

$$x = y - \lambda a$$

substitute into $ax = b$: $\quad a'(y - \lambda a) = b$

$$a'y - \lambda a'a = b$$

$$\lambda = \frac{a'y - b}{a'a}$$

sub into $x = y - \lambda a$ $\quad : x = y - a\frac{a'y - b}{\|a\|}$

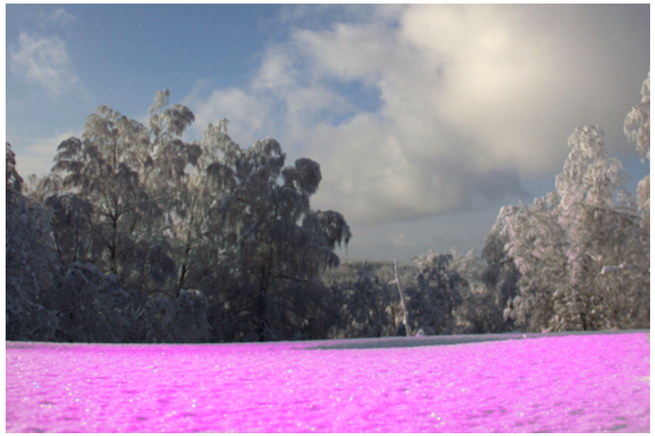
negate: $\quad x = y + a\frac{b - a'y}{\|a\|}$

edit propagation

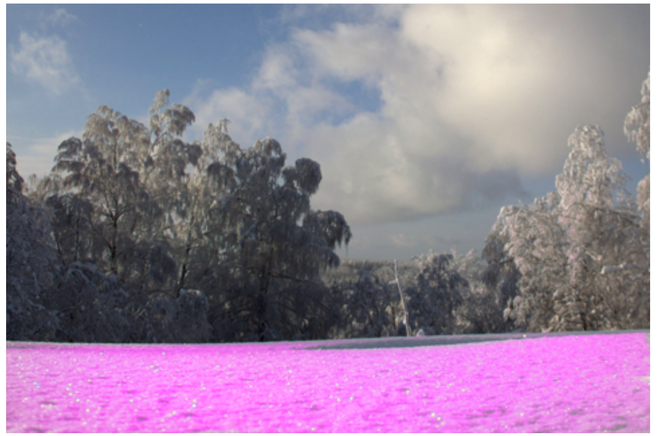
(a) 400 Gabor filters, without random projection

(b) Dimensionality reduced to 40 by random projection

edit propagation



225 dimensions



randomly projected to 30

Generalize

- nonlinear
- unseen data

$$\mathbf{R}\mathbf{w} = \mathbf{z} \qquad \text{standard system}$$

$$\mathbf{P}\mathbf{P}^T\mathbf{w} = \mathbf{z} \qquad \text{substitute approximation}$$

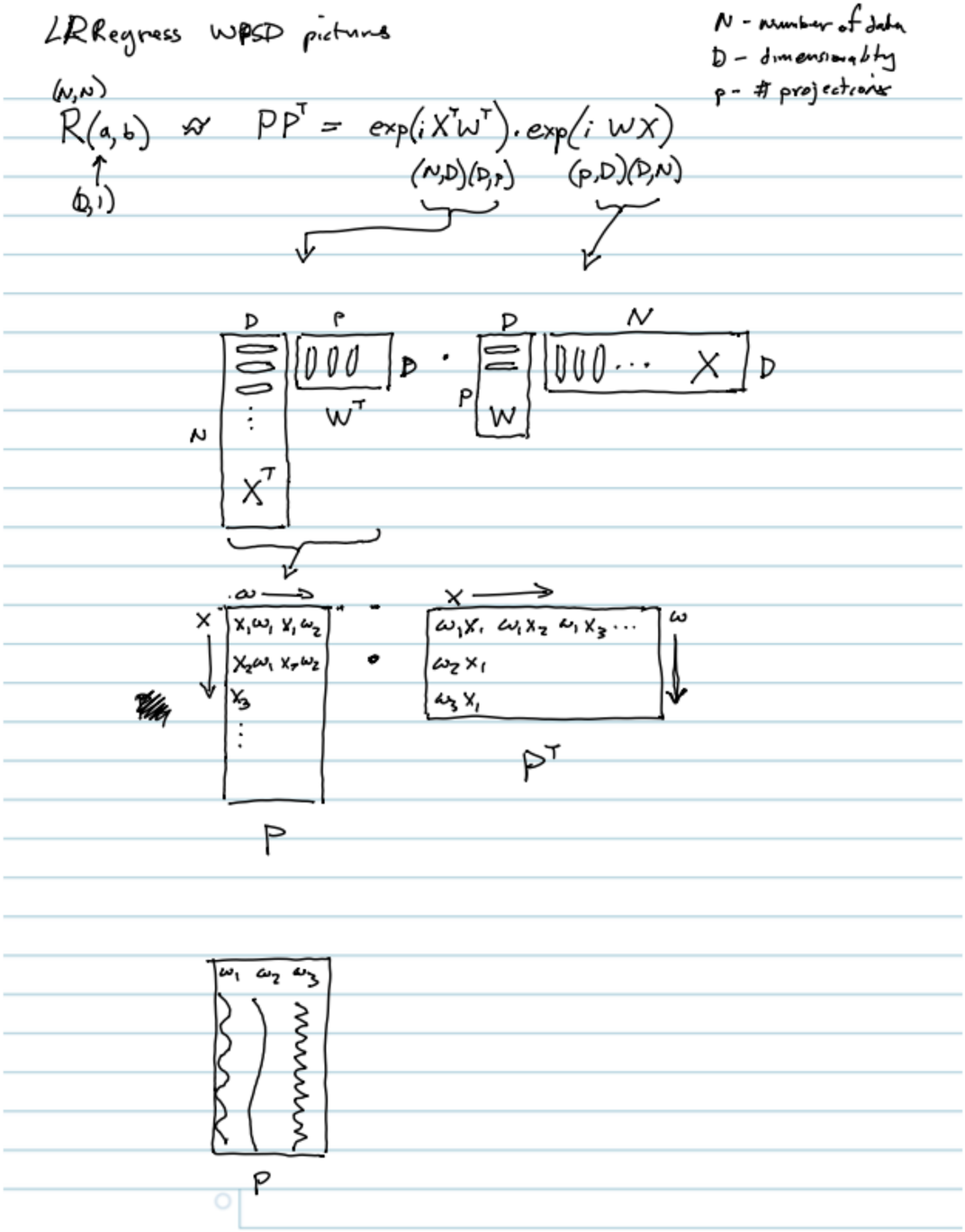
$$\mathbf{P}^T\mathbf{P}(\mathbf{P}^T\mathbf{w}) = \mathbf{P}^T\mathbf{z} \qquad \text{premultiply by } \mathbf{P}^T$$

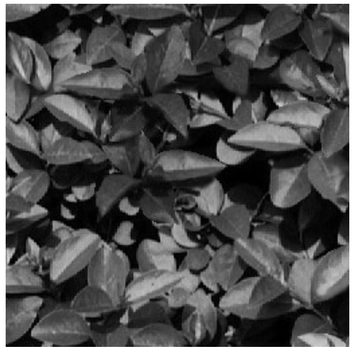
$$\mathbf{P}^T\mathbf{P}\mathbf{v} = \mathbf{P}^T\mathbf{z} \qquad \mathbf{v} \equiv \mathbf{P}^T\mathbf{w}$$

$$\begin{aligned}
\hat{z_0} &= \sum_k^N w_k k(\mathbf{z}_0, \mathbf{z}_k) \\
&\approx \sum_k^N w_k \sum_j^\rho P_{0j} P_{kj} \\
&= \sum_j^\rho P_{0j} \sum_k^N w_k P_{kj} \\
&= \mathbf{p}(\mathbf{z}_0, \cdot)\mathbf{P}^T\mathbf{w}
\end{aligned}$$

$$k(\mathbf{d}) = \int K(\boldsymbol{\omega}) \exp(i2\pi\mathbf{d} \cdot \boldsymbol{\omega}) d\boldsymbol{\omega}$$

$$k(\|\mathbf{z}_0 - \mathbf{z}_k\|) = \int K(\boldsymbol{\omega}) \exp(i2\pi(\mathbf{z}_0 - \mathbf{z}_k) \cdot \boldsymbol{\omega}) d\boldsymbol{\omega}$$

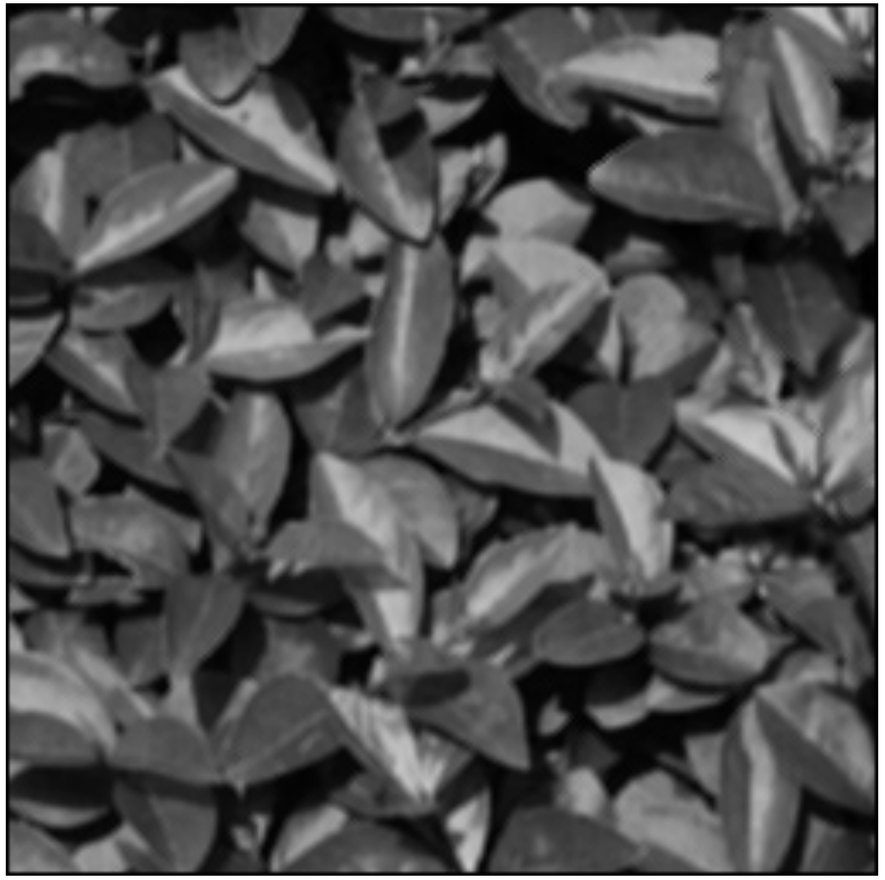


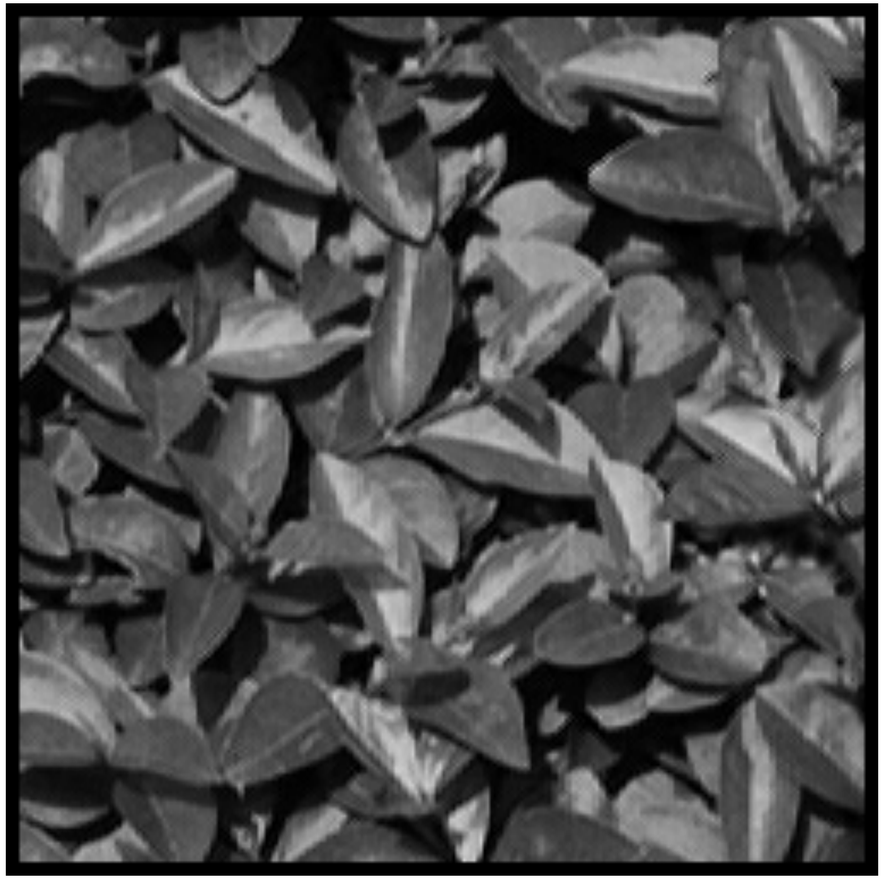


training

original

superresolution





Probable and Improbable Faces

J.P. Lewis, Zhenyao Mo, Ken Anjyo, Taehyun Rhee

Mathematics of Expressive Image Synthesis (MEIS) 2013

Thank you

Resources

- book preprint on Arxiv: Mahoney, Randomized algorithms for matrices and data