ANALYZING REPRESENTATION AND TRANSFORMATIONS

- Reminder: Spafford's lab Mar 3, B1 172, 11:30a

Definitions

- A *representational space* is a specific kind of vector space.
- Vectors are simply collections of mathematical objects (numbers, functions, or other vectors)
- A *vector space* is a set of vectors that is *closed* under addition and multiplication (i.e., a sum or product of any vectors in the set is also in the set)
- A *basis* is an *independent* set of vectors that *span* the vector space

Definitions: independent

- A set of vectors $\mathbf{x}_n$ is independent if

$$a_1\mathbf{x}_1 + a_2\mathbf{x}_2 + \ldots + a_n\mathbf{x}_n = 0$$

- only when

$$a_1 = a_2 = \ldots = a_n = 0$$

- where a_n are scalars.
- So, n must be equal to the dimension of the vectors in $\mathbf{V}$ in order for $\mathbf{x}_n$ to be independent.

Definitions: span

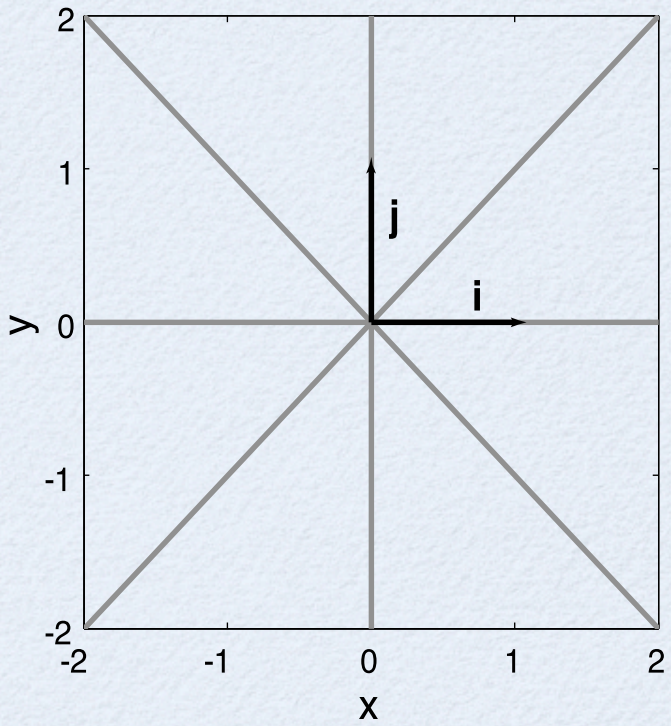
- A set of vectors *spans* a vector space if any vector in that space can be written as a linear sum of those vectors.
- That is, if for all $\mathbf{x} \in \mathbf{V}$ there are some a_n,

$$a_1\mathbf{x}_1 + a_2\mathbf{x}_2 + \ldots + a_n\mathbf{x}_n = \mathbf{x}$$

- then the set of vectors $\mathbf{x}_n$ span the vector space $\mathbf{V}$.

Example

- The standard *Cartesian basis* in any number of dimensions is comprised of the unit vectors along the principle axes (obviously independent and spanning **V**).



Definitions: orthonormal

- Note also, the vectors in this basis are orthogonal, i.e., the dot product of any two will be zero; i.e.,

$$\mathbf{x} \cdot \mathbf{y} = \langle \mathbf{xy} \rangle_n = \sum_n x[n]y[n] = 0$$

- If all the vectors in a basis are orthogonal, it is called an orthogonal basis
- Further, if the length of all the vectors in an orthogonal basis is equal to one then it is an orthonormal basis: the Cartesian basis is such

Overcomplete basis

- If we relax the constraint that the vectors have to be independent, we have what is called an *overcomplete* basis (or sometimes `frame').
  - overcomplete bases are redundant (hence not as succinct) for defining the vector space
  - in noisy, physical systems, this redundancy can prove invaluable for error correction and the efficient use of available resources

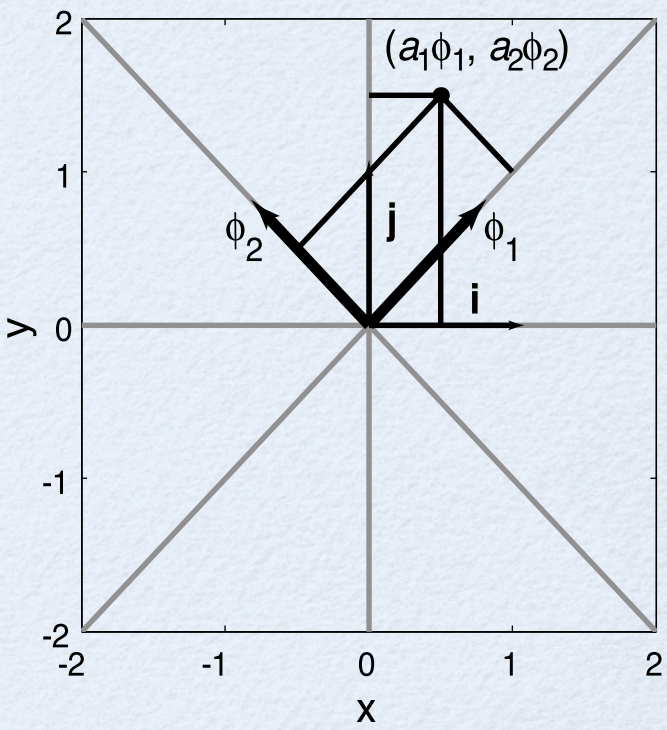
A longer example

- Let the vector **x** be written in a standard Cartesian basis $\mathbf{x} = [x_1, x_2] = x_1\mathbf{i} + x_2\mathbf{j}$
- Suppose we have a different basis, rotated by 45 degrees (ϕ_1, ϕ_2)
- 'Encoding' $a_i = \langle \mathbf{x}\phi_i \rangle_n$
- Projection onto the new basis

$$\mathbf{a} = [a_1, a_2] = a_1\phi_1 + a_2\phi_2$$

- 'Decoding'

$$\mathbf{x} = \sum_i a_i \phi_i$$

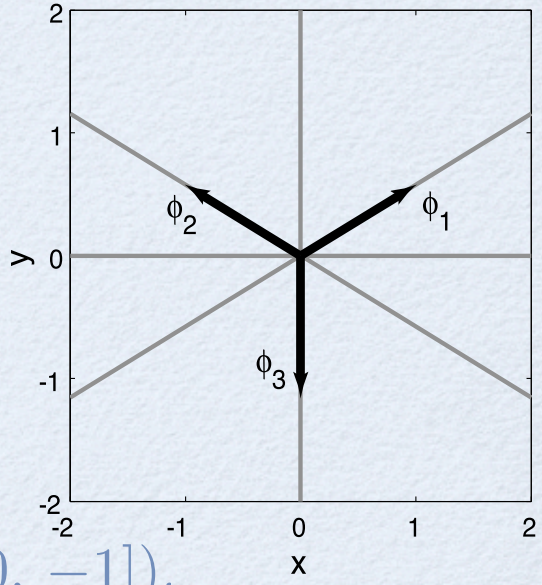


Comments

- This way, we can move back and forth between orthonormal bases.
- Notice that if we substitute the encoding into the decoding, we recover the same vector we originally encoded
- Thus we can think of the coefficients a_i as 'representing', or 'carrying the same information', or 'encoding' the original coefficients x_i

Overcomplete representation

- Suppose we do not know what the decoding basis is, but we know the encoding basis
- We can guarantee that the encoding basis is overcomplete by using redundant, non-orthogonal encoders
- Let's choose the encoding basis as equally spaced at 120 degree intervals (i.e., $\tilde{\phi}_1 = [\frac{\sqrt{3}}{2}, \frac{1}{2}], \tilde{\phi}_2 = [-\frac{\sqrt{3}}{2}, \frac{1}{2}],$ and $\tilde{\phi}_3 = [0, -1]$).



Finding decoders

- We need to identify the set of vectors that span the space into which our vector is being encoded (i.e., the decoding basis).
- To find this basis, we first note, as before, that

$$\mathbf{x} = \sum_i a_i \boldsymbol{\phi}_i$$

- and (notice the tilde! unlike earlier)

$$a_i = \left\langle \mathbf{x} \tilde{\boldsymbol{\phi}}_i \right\rangle_n$$

Finding decoders

- Substitute the encoding into decoding to give

$$\mathbf{x} = \sum_i \left\langle \mathbf{x}\tilde{\boldsymbol{\phi}}_i \right\rangle_n \boldsymbol{\phi}_i$$

- Writing the dot product explicitly, we get x[m]

$$\begin{aligned} x[m] &= \sum_{i,n} x[n]\tilde{\phi}_i[n]\phi_i[m] \\ &= \sum_n x[n] \sum_i \tilde{\phi}_i[n]\phi_i[m] \end{aligned}$$

- So,

$$\delta_{nm} = \sum_i \tilde{\phi}_i[n]\phi_i[m]_i$$

Finding decoders

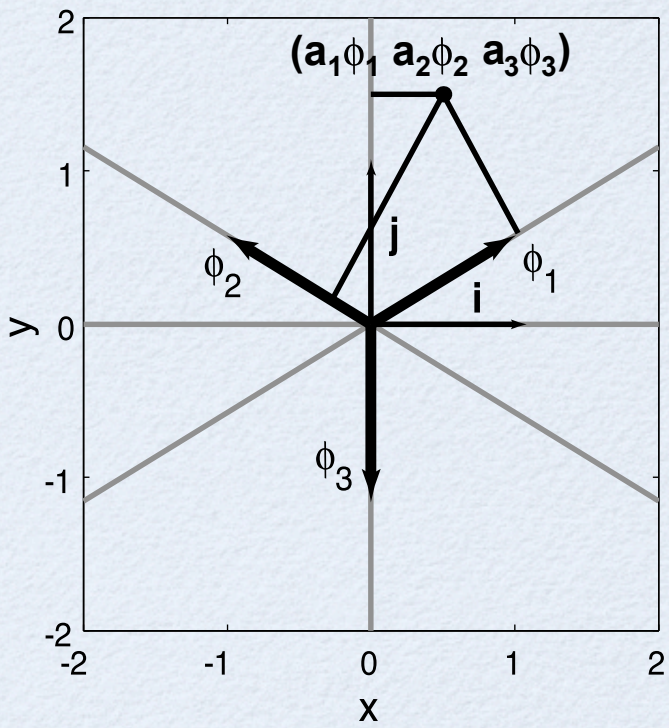
- In matrix notation, $\mathbf{I} = \tilde{\phi}\phi$
- Solving for the decoders

$$\begin{aligned} \mathbf{I} &= \tilde{\phi}\phi \\ \tilde{\phi}^T &= \tilde{\phi}^T\tilde{\phi}\phi, \\ \phi &= \left(\tilde{\phi}^T\tilde{\phi}\right)^{-1}\tilde{\phi}^T \end{aligned}$$

- Calculating gives

$$\phi_1 = [\frac{\sqrt{3}}{3}, \frac{1}{3}], \phi_2 = [-\frac{\sqrt{3}}{3}, \frac{1}{3}], \text{ and } \phi_3 = [0, -\frac{2}{3}]$$

- That is, $\phi_i = \frac{2}{3}\tilde{\phi}_i$



Comments

- The ϕ_i are an *overcomplete basis* for the vector space in which the a_i are coordinates. Hence are the decoding vectors defining the representation
- A set of encoders/decoders is like this is called biorthogonal because together they act as an orthogonal basis.
- Technically, the NEF encoders/decoders don't satisfy the definitions provide here, but they are close

Basis functions

- Applies equally to basis *functions*
- Just as basis vectors define a vector space, so basis functions define a function space
- E.g. sines and cosines for a Fourier decomposition. This is an orthonormal basis.
- Can have overcomplete basis functions as well.
- The tuning curves are like a set of overcomplete basis functions for the space they represent.

Decomposing Γ

- Consider the noise free case

$$\gamma_{ij} = \langle a_i(\mathbf{x}) a_j(\mathbf{x}) \rangle_{\mathbf{x}}$$

- This is called a Gram matrix (a correlation matrix without the means removed). Measures similarity of all neural responses.
- Tells us about the representation of the vector space by the neurons

Decomposing Γ

- Construct an 'activity matrix'

$$\mathbf{A}^T = \begin{bmatrix} a_1(\mathbf{x}_{min}) & a_1(\mathbf{x}_{min} + \Delta_{\mathbf{x}}) & \cdots & a_1(\mathbf{x}_{max} - \Delta_{\mathbf{x}}) & a_1(\mathbf{x}_{max}) \\ a_2(\mathbf{x}_{min}) & & & & a_2(\mathbf{x}_{max}) \\ \vdots & \vdots & \cdots & \vdots & \vdots \\ a_{N-1}(\mathbf{x}_{min}) & & & & a_{N-1}(\mathbf{x}_{max}) \\ a_N(\mathbf{x}_{min}) & a_1(\mathbf{x}_{min} + \Delta_{\mathbf{x}}) & \cdots & a_1(x_{max} - \Delta_{\mathbf{x}}) & a_N(\mathbf{x}_{max}) \end{bmatrix}$$

Decomposing Γ

- So we can write our estimate

$$\hat{\mathbf{X}}_{N_\Delta \times N_v} = \mathbf{A}_{N_\Delta \times N} \boldsymbol{\phi}_{N \times N_v}$$

- Need the optimal decoders, so take $\mathbf{X}$ as given (i.e., the actual domain)

$$\begin{aligned} \mathbf{X} &= \mathbf{A}\boldsymbol{\phi} \\ \mathbf{A}^T\mathbf{X} &= \mathbf{A}^T\mathbf{A}\boldsymbol{\phi} \\ (\mathbf{A}^T\mathbf{A})^{-1}\mathbf{A}^T\mathbf{X} &= \boldsymbol{\phi}. \end{aligned}$$

Decomposing Γ

- Taking the inverse correctly results in linear optimal decdoders, we can write

$$\begin{aligned} \gamma &= \mathbf{A}^T\mathbf{A} \\ \Upsilon &= \mathbf{A}^T\mathbf{X} \end{aligned}$$

- In other words,

$$\begin{aligned} \phi &= \gamma^{-1}\Upsilon \\ &= (\mathbf{A}^T\mathbf{A})^{-1}\mathbf{A}^T\mathbf{X} \end{aligned}$$

- Which means

$$\begin{aligned} \hat{\mathbf{X}} &= \mathbf{A}\phi \\ &= \mathbf{A}(\mathbf{A}^T\mathbf{A})^{-1}\mathbf{A}^T\mathbf{X}. \end{aligned}$$

SVD

- How do we take the inverse correctly?
- Because there is no noise, and tuning curves are very similar, the matrix is likely to be singluar
- Use singular value decomposition (SVD) to decompose and approximately invert the matrix.
- SVD decomposition of an *M*x*N* matrix, **B**, gives

$$\mathbf{B}_{M\times N} = \mathbf{U}_{M\times N}\mathbf{S}_{N\times N}\mathbf{V}^{T}_{N\times N}$$

- **S** is a diagonal matrix of singular values

SVD

- When $\mathbf{B}$ is square and symmetrical (as for $\boldsymbol{\gamma}$), this simplifies to $\boldsymbol{\gamma} = \mathbf{U}\mathbf{S}\mathbf{U}^T$
- Also, $\boldsymbol{\gamma}^{-1} = \mathbf{U}\mathbf{S}^{-1}\mathbf{U}^T$
- When $\boldsymbol{\gamma}$ is singular (or nearly so), some elements of $\mathbf{S}$ are zero (or very small), so the inverse of $\mathbf{S}$ includes infinite (or very large) terms
- The SVD `pseudo-inverse' is defined where for S_i=0, the inverse is set to 0 (∞=0!).

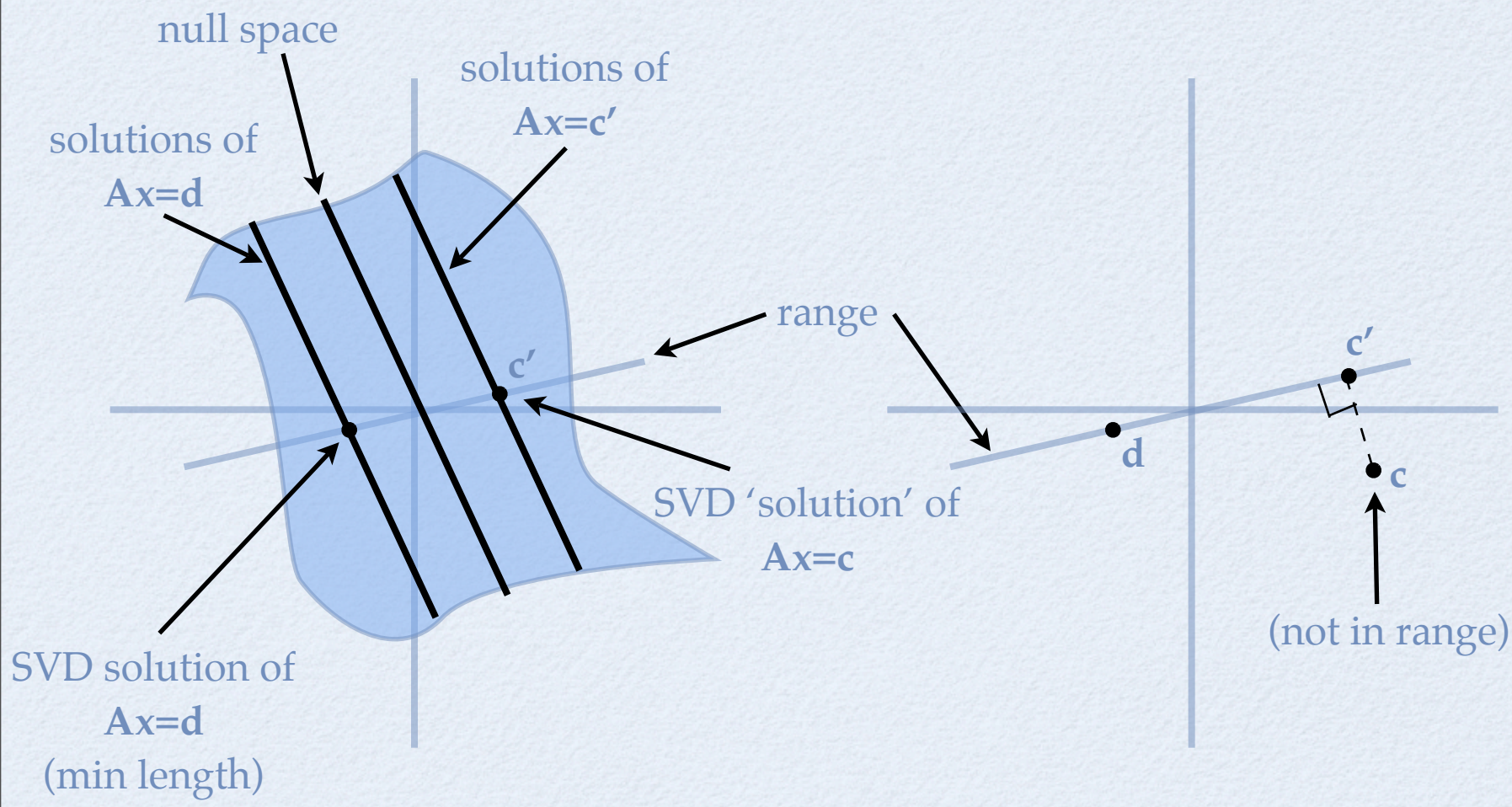
SVD

- SVD can be very informative:
  - The columns of **U** whose corresponding singular values are non-zero form an orthonormal basis that spans the range of γ
  - The columns of **U** whose corresponding SVs are zero form an orthonormal basis that spans the null space (i.e., $\mathbf{x}$ s.t. $\gamma\,\mathbf{x}=\mathbf{0}$).

SVD

- When a vector in Υ lies in the range of γ, the SVD pseudo-inverse guarantees that the corresponding vector from ϕ minimizes the length of that ϕ vector.
- Given that γ is singular, there are an infinite number of solutions for ϕ. The solution that provides the shortest vector is a natural choice from the set

SVD (singular matrix)



SVD

- When a vector in Υ lies outside the range of γ, the SVD pseudo-inverse guarantees that the best (in the least squares sense) ϕ given Υ is found.
- I.e., this `pseudo-inverse' minimizes the error. Thus, we can use SVD to find the optimal decoding functions, which we can now write as

$$\phi = \mathbf{U}\mathbf{S}^{-1}\mathbf{U}^T\mathbf{A}^T\mathbf{X}$$

- Which is gives the same ϕ as always.

Possible transformations

- Let's do the same thing to function decoders

$$\begin{aligned} f(\mathbf{X}) &= \mathbf{A}\boldsymbol{\phi}^f \\ \mathbf{A}^T f(\mathbf{X}) &= \mathbf{A}^T\mathbf{A}\boldsymbol{\phi}^f \\ \boldsymbol{\phi}^f &= (\mathbf{A}^T\mathbf{A})^{-1}\mathbf{A}^T f(\mathbf{X}) \\ &= \boldsymbol{\gamma}^{-1}\boldsymbol{\Upsilon} \end{aligned}$$

- So we can find lease-squares optimal decoders using

$$\boldsymbol{\phi}^f = \mathbf{U}\mathbf{S}^{-1}\mathbf{U}^T\mathbf{A}^T f(\mathbf{X})$$

Possible transformations

- So, the representational decoder, is found in the special case where $f(\mathbf{x})=\mathbf{x}$.
- Regardless of which transformation we need decoders for, we always perform SVD on the same matrix, $\gamma=\mathbf{A}^T\mathbf{A}$.
- Understanding the properties of γ can provide insight into all possible decodings of the population, a_i.

Possible transformations

- SVs tell us the importance of the corresponding **U** vector. Importance being:
- related to the error that would result if we left a particular vector out of the mapping
- related to the variance of population firing along the vectors in the γ matrix.
- the amount of (independent) information about population firing that can be extracted by looking only at data projected onto that **U** vector.

Possible transformations

- In general, we can think of the magnitude of the singular value as telling us how relevant the dimension defined by the corresponding **U** vector is to the identity of the matrix we have decomposed.
- Since the matrix we have decomposed is like the correlation matrix of the neuron tuning curves, the large singular values are most important for accounting for the structure of those correlations.

Possible transformations

- Notice that the vectors in **U** are orthogonal:
  - they provide an (ordered) orthogonal basis for that matrix, where the original γ matrix was generated by a non-ordered non-orthogonal basis (the neuron tuning curves)
  - Consider a point in 'neuron space'

$$\mathbf{a} = a_1\mathbf{e}_1 + a_2\mathbf{e}_2 + \ldots + a_N\mathbf{e}_N$$

  - Here, $\mathbf{e}_i$ are axis vectors, one for each neuron with activity a_i.

Possible transformations

- Because the neural responses are non-independently driven by some variable, **x**, only a subspace of the space spanned by the $\mathbf{e}_i$ vectors is ever actually occupied by the population.
- The γ matrix, because it tells us the correlations between all neurons in the population, provides us with the information we need to determine what that subspace is (i.e., **U**)

Possible transformations

- Let's do some math:

$$\hat{\mathbf{X}} = \mathbf{A}\mathbf{U}\mathbf{S}^{-1}\mathbf{U}^T\mathbf{A}^T\mathbf{X}$$

- Or more simply $\hat{\mathbf{X}} = \chi\Phi$
- So, $\chi = \mathbf{A}\mathbf{U}$
- or $\chi_m(\mathbf{x}) = \sum_i a_i(\mathbf{x})U_{im}$

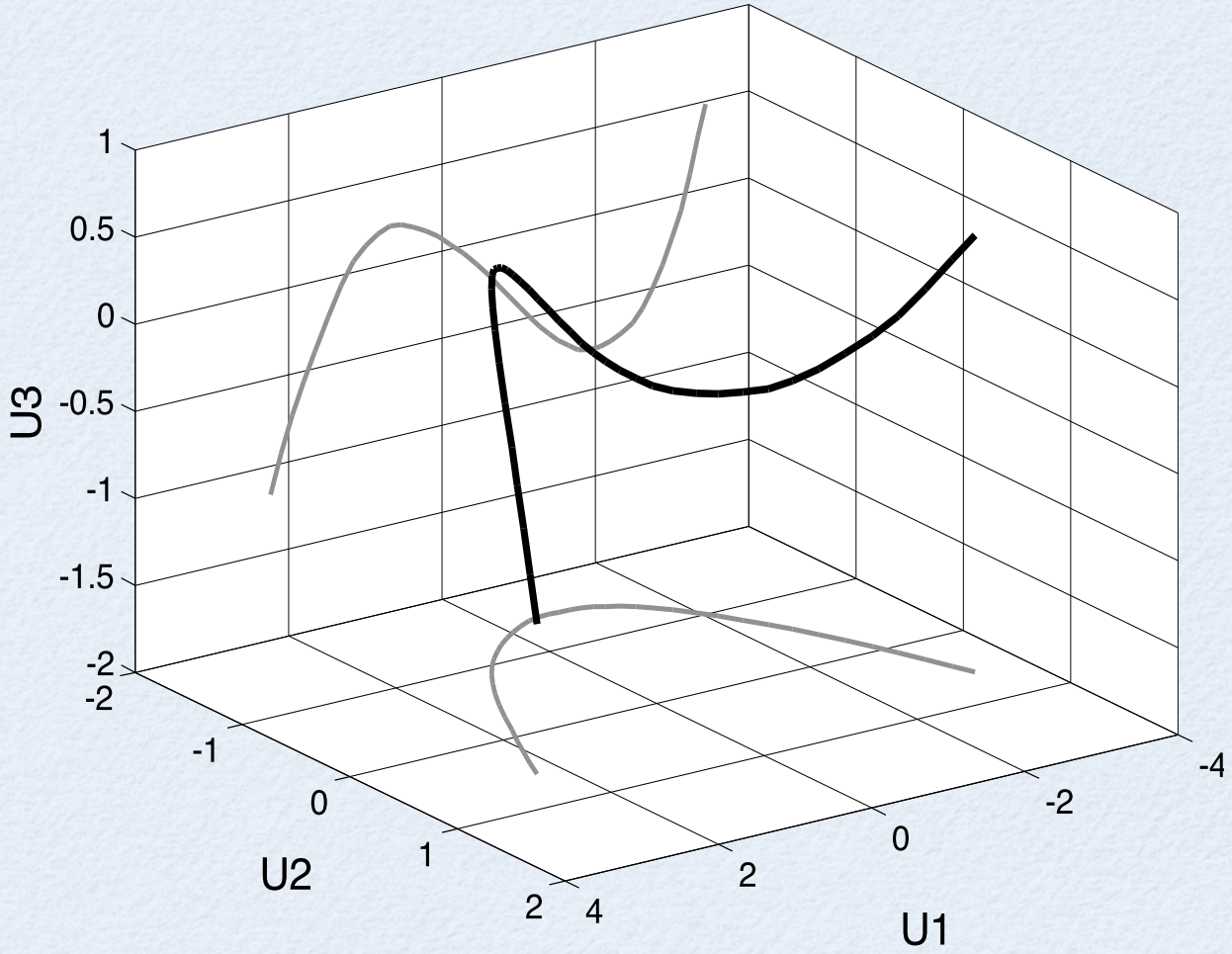
Possible transformations

- and continuing
$$\begin{aligned}\Phi &= \mathbf{S}^{-1}\mathbf{U}^T\mathbf{A}^T\mathbf{X} \\ &= \mathbf{U}^T\mathbf{U}\mathbf{S}^{-1}\mathbf{U}^T\mathbf{A}^T\mathbf{X} \\ &= \mathbf{U}^T\boldsymbol{\phi}\end{aligned}$$
- so
$$\Phi_m = \sum_i U_{mi}\phi_i$$
- Notice that χ and $\mathbf{\Phi}$ are rotated versions of $\mathbf{A}$ and ϕ respectively. They are rotated into the coordinate system defined by $\mathbf{U}$.

Possible transformations

- We can think of **U** as the rotation matrix that aligns the first axis of the coordinate system along the dimension with the greatest variance in the encoding of **x**, the second axis along the dimension with the second greatest variance, etc.
- So the χ vectors also end up being orthogonal and ordered by importance
- So they are basis functions for the function space computable from **A**

SVD 3D projections

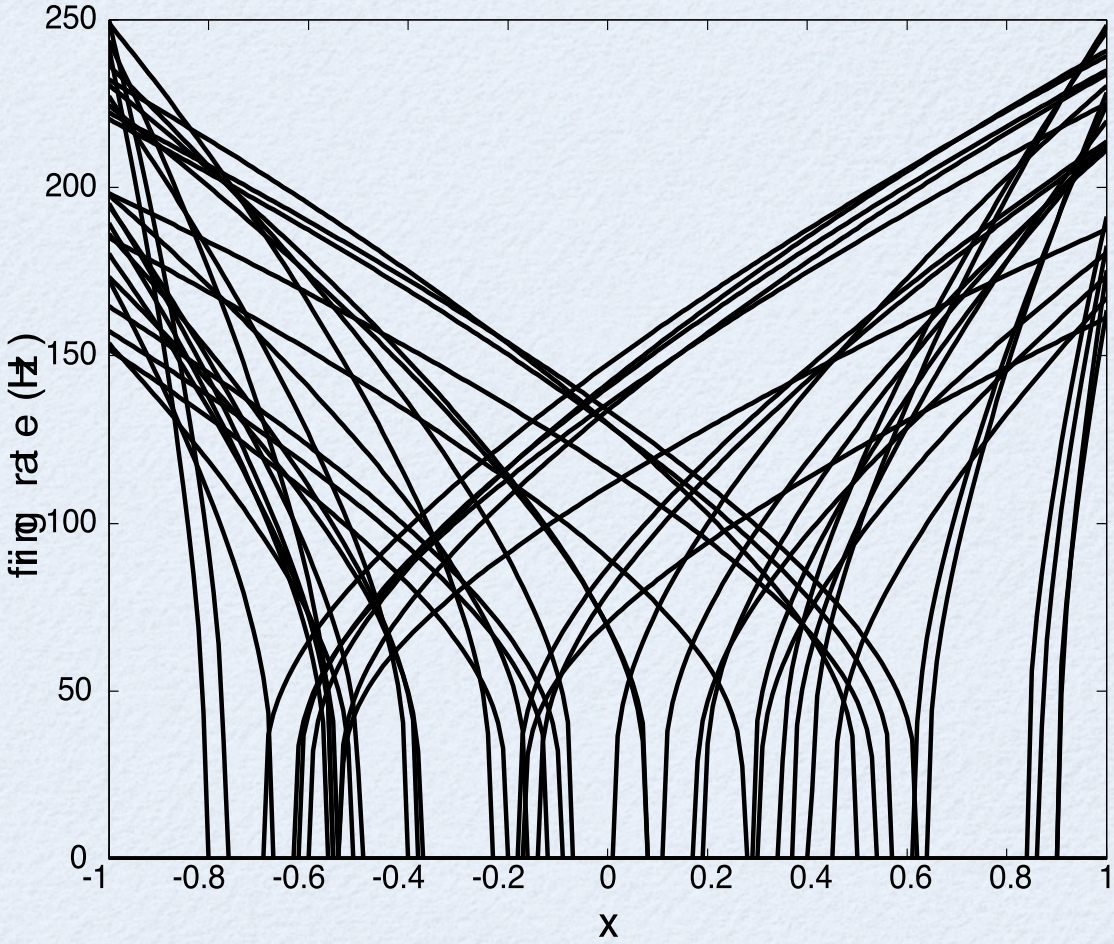


Possible functions

- Whichever $\chi(\mathbf{x})$ functions have reasonably large associated singular values, are exactly the functions that we can do a good job of extracting from our encoding of the input space
- We can also extract any linear combinations of those functions quite well.
- Because these functions are ordered, the more useful the `first' function is for reconstructing some transformation, $f(\mathbf{x})$, the better we can extract it

Possible functions

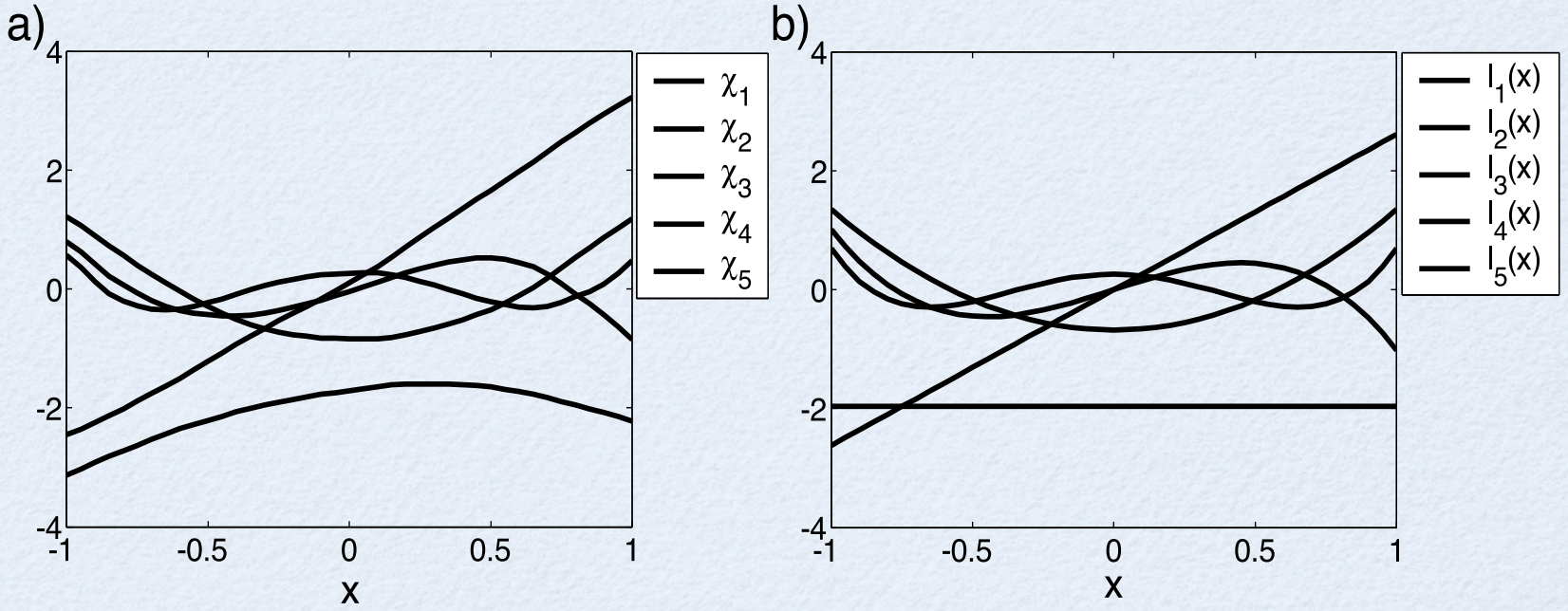
- Let's apply this analysis. Start with broad tuning.



Possible functions

- We get the following (normalized) functions. (b) is the Legendre polynomials

$$l_i(x) = \frac{(-1)^i}{2^i i!} \frac{d^i}{dx^i} \left[\left(1 - x^2\right)^i \right]$$



Possible functions

- The similarity between $\chi(x)$ and $l_i(x)$ means that this neural population supports the extraction of functions that can be well-estimated using the standard Legendre basis.
- But the $\chi(x)$ functions are ordered by their singular values. Thus, the higher-order polynomial terms are not as well encoded by our population as the lower-order ones

Comments

- Tuning curves are very broad, and the polynomial basis is also very broad
- The l_i basis is ordered by decreasing linearity so we expect the functions in precisely that order
- None of this would be true if the population did not evenly tile the input space.
- The basis does not just depend on the general `shape' of the neuron tuning curves, but also on the 'heterogeneity' of the population

Higher dimensions

- n-dimensional vectors have cross terms in the computable functions

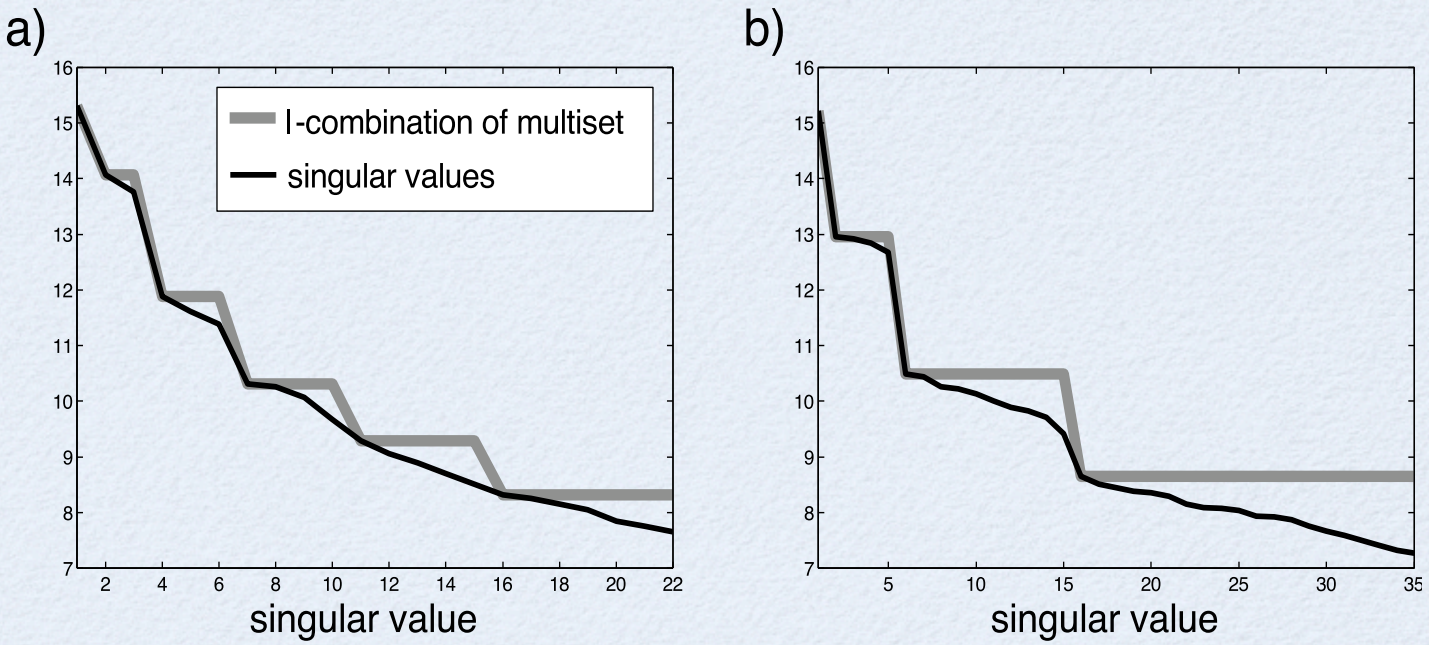
$$f(\mathbf{x}) = c_0 + c_1 x_1 + c_2 x_2 + c_3 x_1^2 + c_4 x_2^2 + c_5 x_1 x_2 + \ldots$$

$$= \sum_{l=0}^{N_{order}} \sum_{n=0}^{l} c_{n,l-n} x_1^n x_2^{l-n}$$

- Lots of cross terms, how big are the sigular values?

Multiset and singular values

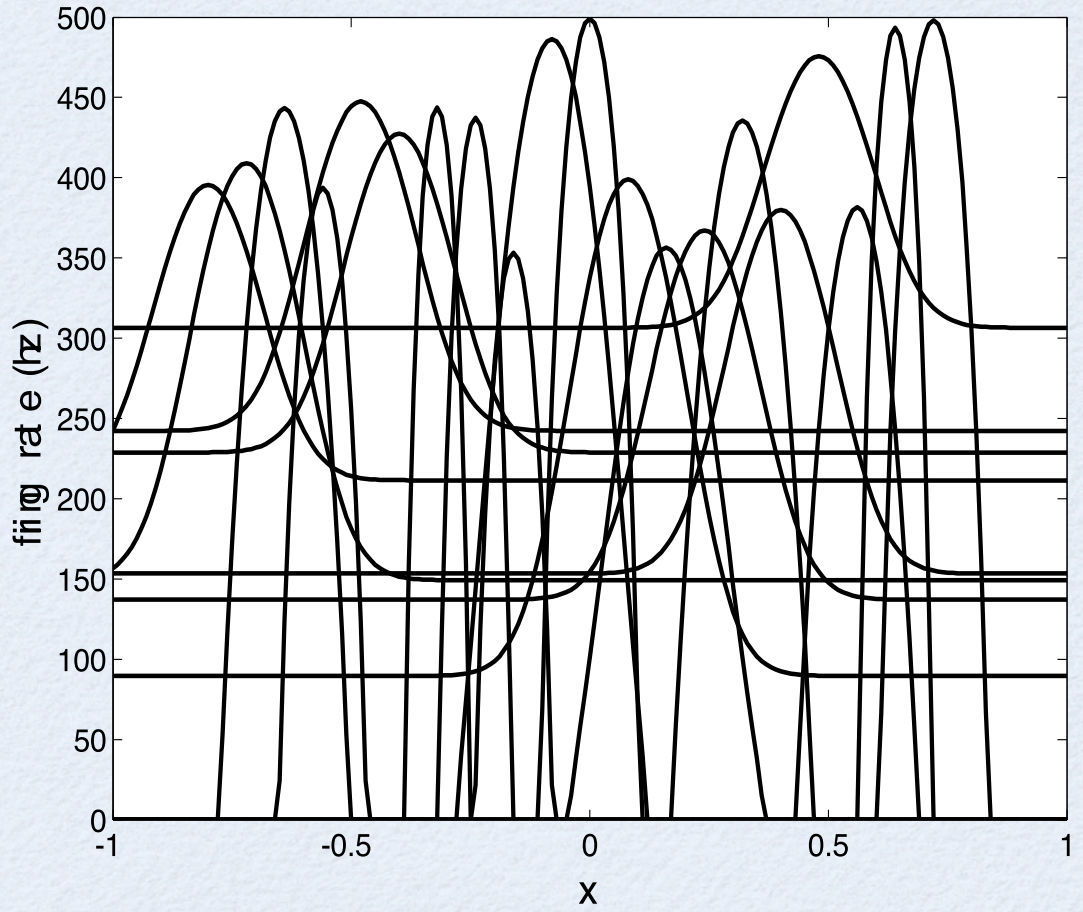
- a) 2D b) 4D grey line: $N_S(l, D) = \frac{(l + D - 1)!}{l!(D - 1)!}$



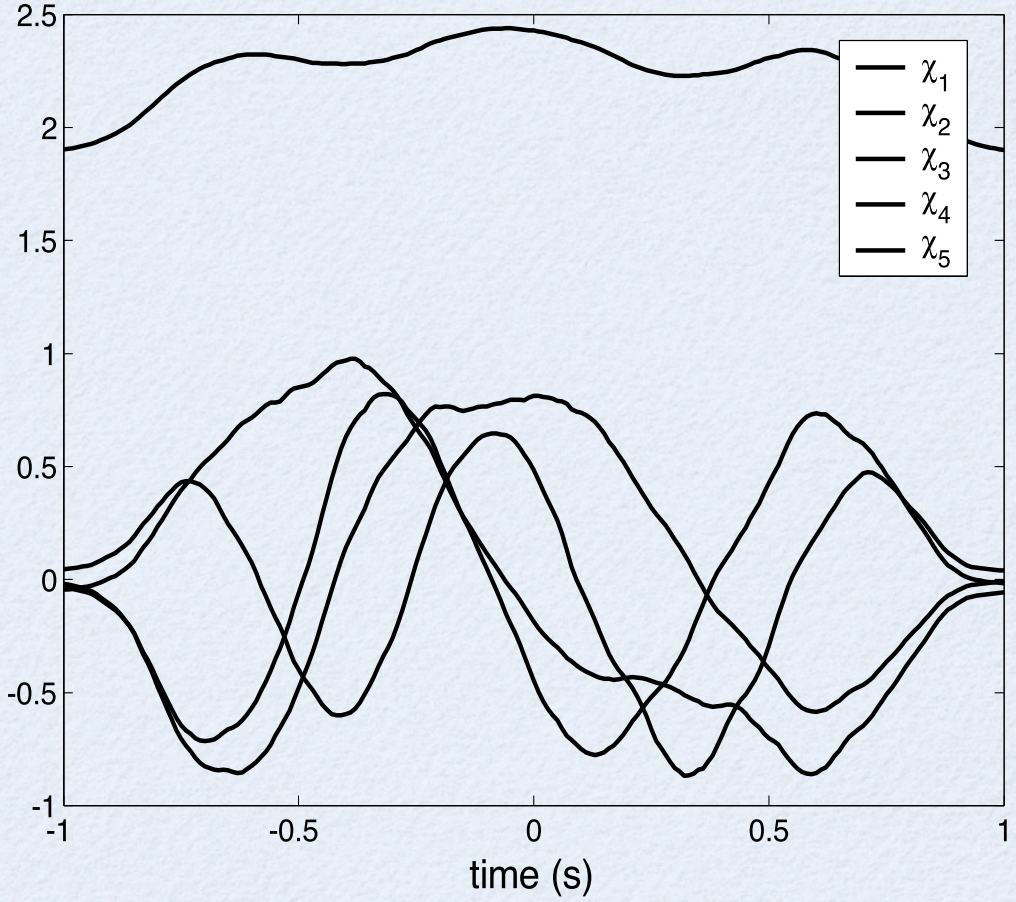
- So, higher-D means higher order is harder.

Gaussian tuning curves

- Same analysis for Gaussian tuning

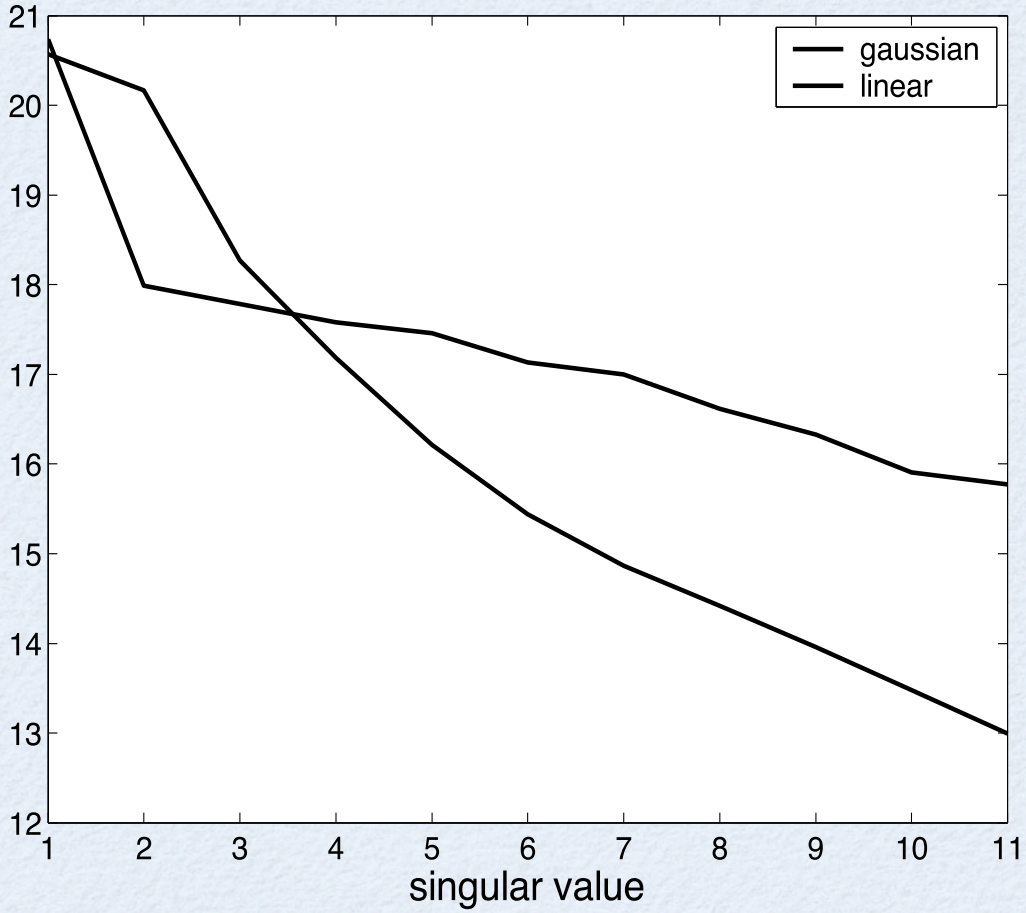


Gaussian basis functions



SVs: Gaussian vs Linear

- Comparing scalar singular values for the two:



Comments

- Gaussian basis looks like a Fourier basis (but pinned because of the end effects)
- The Gaussian basis is better for encoding localized functions (see singular values)
- In general, examining $\chi(x)$ determines what kinds of transformations are and are not well-supported by that population.
- Extremely useful for constraining hypotheses

Noise

- Essentially everything is the same, except the function ‘lines’ become ‘tubes’ of uncertainty
- Importantly, however, the noise doesn't scale with the singular values, it is isometric in the vector space.
- This means that small singular values are more greatly affected by noise.

Error with noise

- Write as before

$$
\begin{aligned}
E &= \left\langle \left[\mathbf{x} - \sum_i \left(a_i(\mathbf{x}) + \eta_i \right) \boldsymbol{\phi}_i \right]^2 \right\rangle_{\mathbf{x},\eta} \\
&= \left\langle \left[\mathbf{x} - \sum_i a_i(\mathbf{x}) \boldsymbol{\phi}_i \right]^2 + \sigma_\eta^2 \sum_i \boldsymbol{\phi}_i^2 \right\rangle_{\mathbf{x}}
\end{aligned}
$$

Noise

- We can now use our SVD expressions to give

$$
\begin{aligned}
E &= \left\langle \left[\mathbf{x} - \sum_m \chi_m \Phi_m \right]^2 + \sigma_\eta^2 \sum_{i,j} \phi_i \delta_{ij} \phi_j \right\rangle_{\mathbf{x}} \\
&= \left\langle \left[\mathbf{x} - \sum_m \chi_m \Phi_m \right]^2 + \sigma_\eta^2 \sum_{i,j} \phi_i \sum_m U_{im} U_{mj} \phi_j \right\rangle_{\mathbf{x}} \\
&= \left\langle \left[\mathbf{x} - \sum_m \chi_m \Phi_m \right]^2 + \sigma_\eta^2 \sum_m \Phi_m^2 \right\rangle_{\mathbf{x}}
\end{aligned}
$$

Noise

- Minimize with respect to rotated decoders:

$$\frac{dE}{d\Phi_n} = \left\langle 2\left[\mathbf{x} - \sum_m \chi_m \Phi_m\right](-\chi_n) + 2\sigma_\eta^2 \Phi_n \right\rangle_{\mathbf{x}}$$

$$0 = -2\left\langle \chi_n \mathbf{x} \right\rangle_{\mathbf{x}} + 2\left\langle \sum_m \chi_m \chi_n \Phi_n \right\rangle_{\mathbf{x}} + 2\sigma_\eta^2 \Phi_n$$

$$\left\langle \chi_n \mathbf{x} \right\rangle_{\mathbf{x}} = S_n \Phi_n + \sigma_\eta^2 \Phi_n$$

$$\Phi_n = \frac{\left\langle \chi_n \mathbf{x} \right\rangle_{\mathbf{x}}}{S_n + \sigma_\eta^2}$$

Noise

- So, the residual error will be

$$
\begin{aligned}
E_r &= \left\langle [\mathbf{x} - \hat{\mathbf{x}}]^2 \right\rangle_{\mathbf{x},\eta} \\
&= \left\langle \left[\mathbf{x} - \sum_m \chi_m \Phi_m \right]^2 \right\rangle_{\mathbf{x},\eta} \\
&= \langle \mathbf{x}^2 \rangle_{\mathbf{x},\eta} - 2 \left\langle \mathbf{x} \sum_m \chi_m \Phi_m \right\rangle_{\mathbf{x},\eta} + \left\langle \sum_m \chi_m \Phi_m \right\rangle^2_{\mathbf{x},\eta}
\end{aligned}
$$

Noise

- Substituting the optimal rotated decoders:

$$
\begin{aligned}
E_r &= \left\langle \mathbf{x}^2 \right\rangle_{\mathbf{x},\eta} - 2 \left\langle \mathbf{x} \sum_m \chi_m \frac{\left\langle \chi_m \mathbf{x} \right\rangle_{\mathbf{x}}}{S_m + \sigma_\eta^2} \right\rangle_{\mathbf{x},\eta} + \ldots \\
&\qquad \left\langle \sum_m \chi_m \frac{\left\langle \chi_m \mathbf{x} \right\rangle_{\mathbf{x}}}{S_m + \sigma_\eta^2} \right\rangle_{\mathbf{x},\eta}^2 \\
&= \left\langle \mathbf{x}^2 \right\rangle_{\mathbf{x}} - 2 \sum_m \frac{\left\langle \chi_m \mathbf{x} \right\rangle_{\mathbf{x}}^2}{S_m + \sigma_\eta^2} + \sum_m (S_m + \sigma_\eta^2) \frac{\left\langle \chi_m \mathbf{x} \right\rangle_{\mathbf{x}}^2}{\left(S_m + \sigma_\eta^2\right)^2} \\
&= \left\langle \mathbf{x}^2 \right\rangle_{\mathbf{x}} - \sum_m \frac{\left\langle \chi_m \mathbf{x} \right\rangle_{\mathbf{x}}^2}{S_m + \sigma_\eta^2}
\end{aligned}
$$

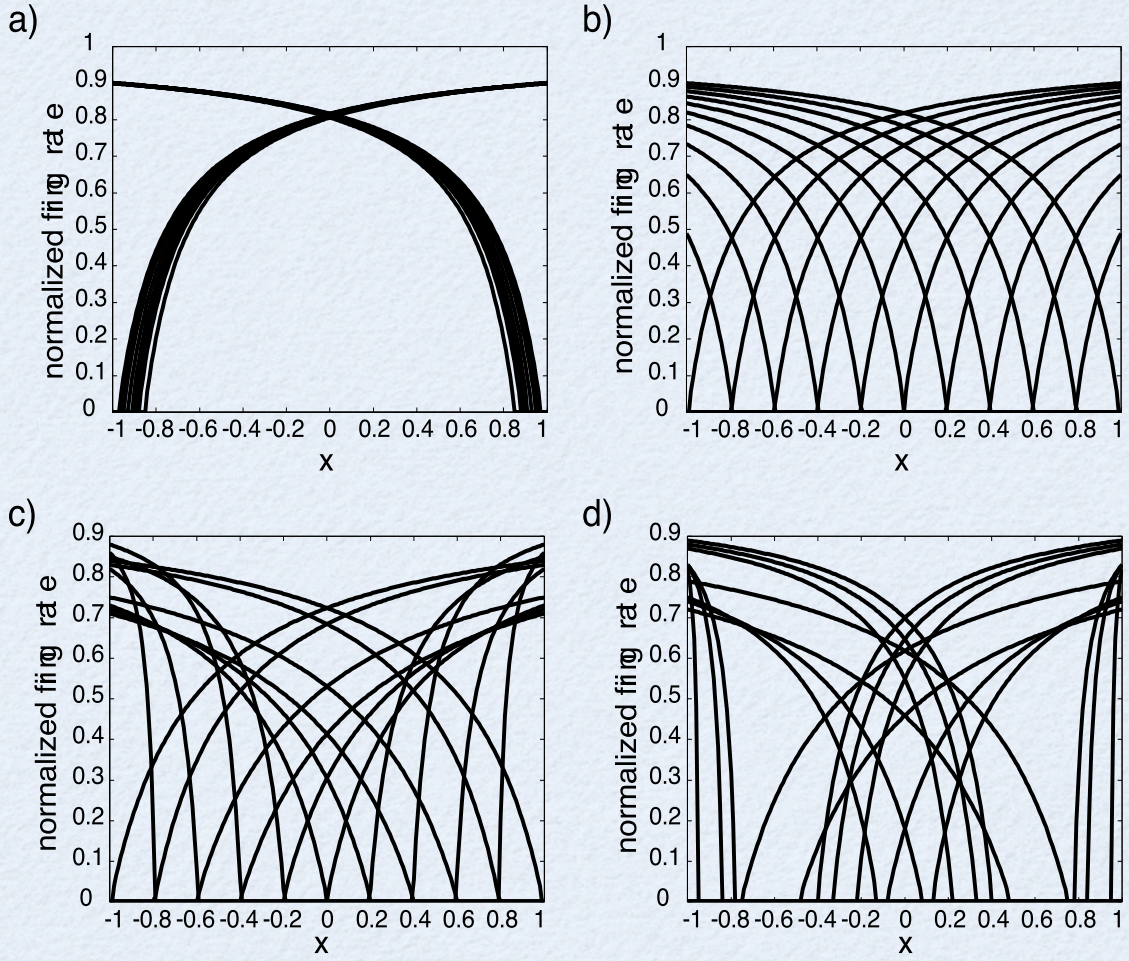
Comments

- Shows how much the *m*th basis function reduces the error in our estimate under noise
- As the singular value approaches the value of the variance of the noise, the corresponding element does not usefully contribute to the representation.
- When the noise becomes near the magnitude of that normalization term (i.e., SNR = 1 or less), the projection becomes `mis-normalized' and thus contributes incorrectly to the representation

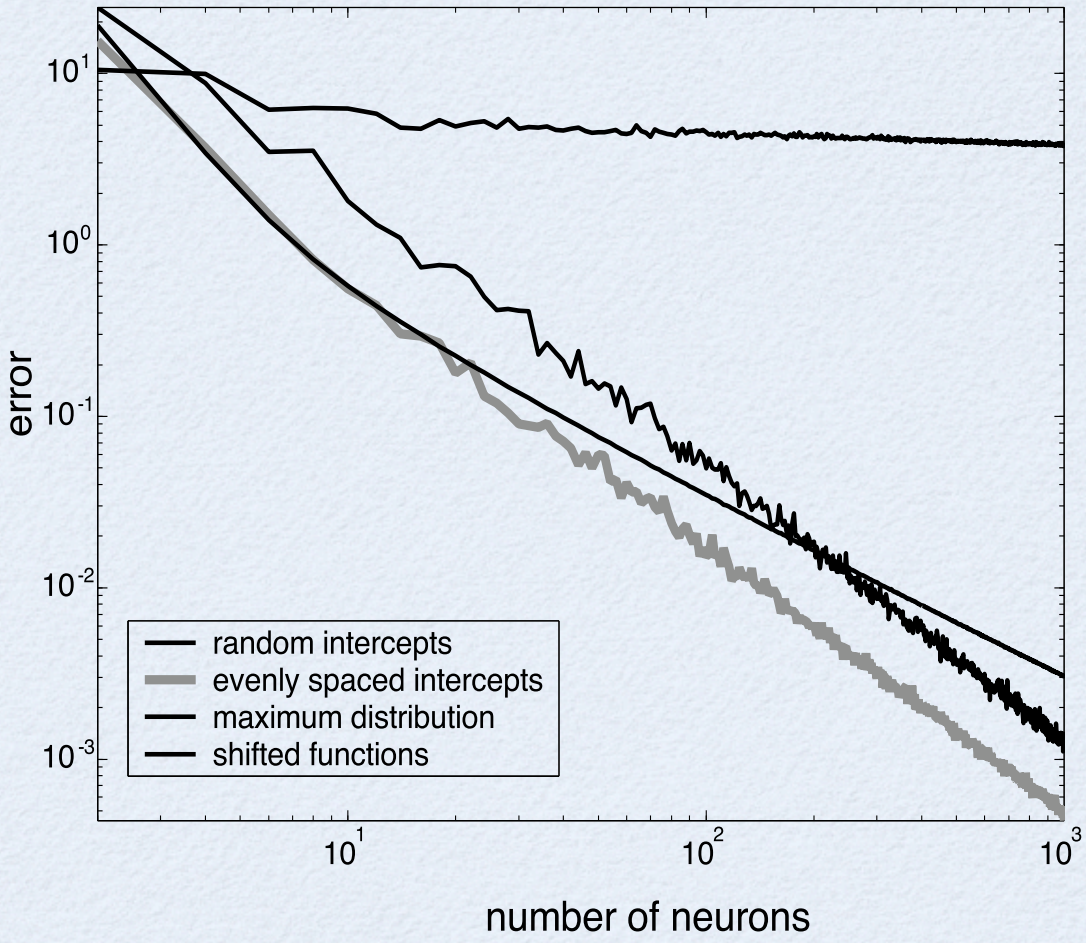
Comments

- So, *those basis functions whose corresponding singular value is equal to or smaller than the noise, should not be used in a good representation*
- So, we can simply `lop off' some of the singular values when doing the inverse to get the same result as including a certain amount of noise in our calculation of $\boldsymbol{\Gamma}$ (more analysis needed)

Heterogeneity



RMS error for distributions



Heterogeneity

- The precise nature of the tuning curves matters.
- In the book we show that heterogeneity is a good balance between tiling and ease of construction
- Heterogeneity provides good reduction of error under noise.
- The representational capacity (number of perfectly represented points) is also high for heterogeneous populations.