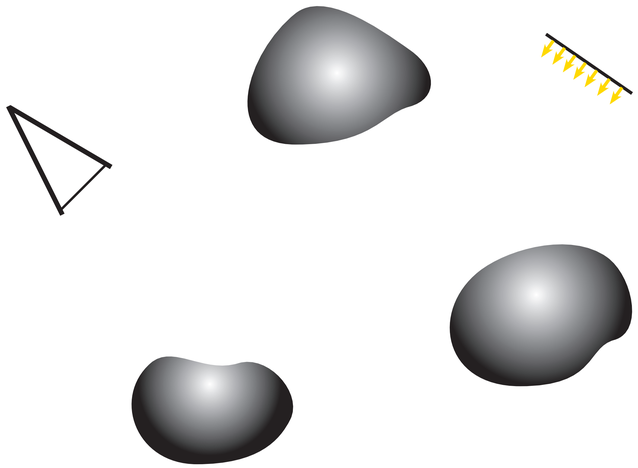
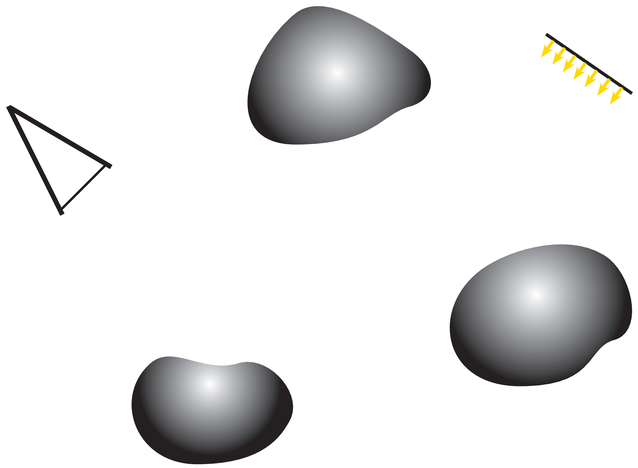
How many rays do we need?

Christian Lessig
Otto-von-Guericke-Universität Magdeburg

Objective

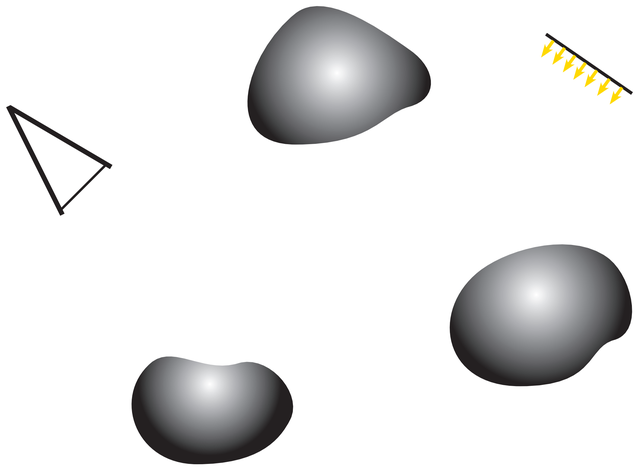


Objective



Rectilinear transport in known scene

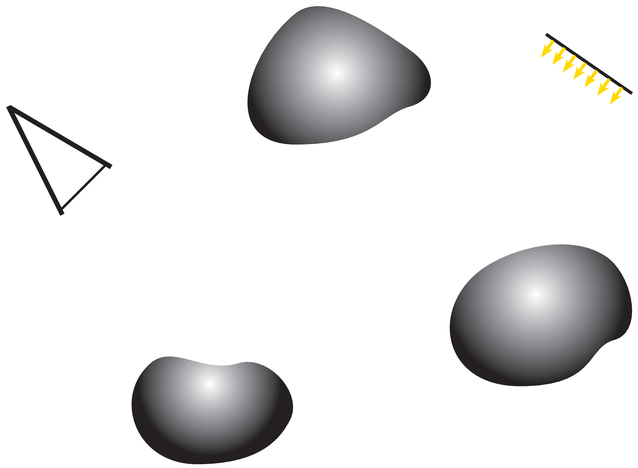
Objective



G

Rectilinear transport in known scene

Objective



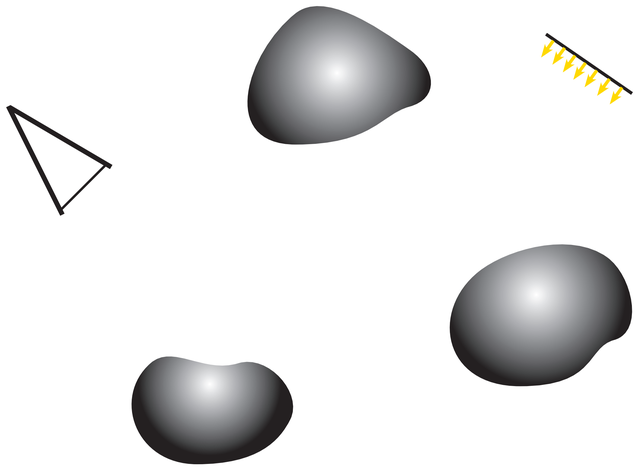
Rectilinear transport in
known scene

G

$\hat{f} \in L_2([0,1]^2)$

$f \in \mathcal{H} \subset L_2, \dim(\mathcal{H}) < \infty$

Objective



Rectilinear transport in known scene

$$\xrightarrow{G}$$

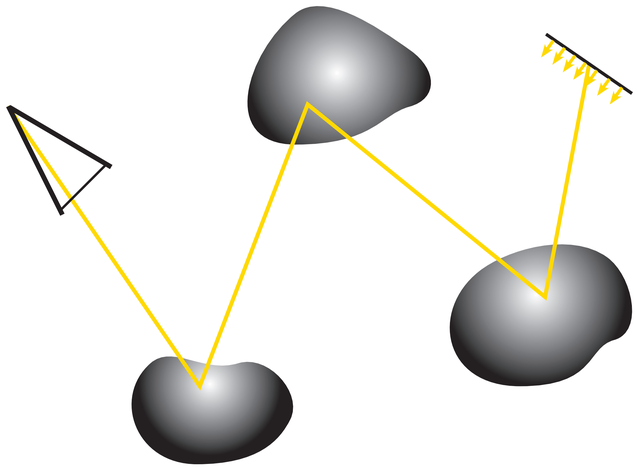
(close-to-) optimal for

$$\|\hat{f} - f\| < \epsilon$$

$$\hat{f} \in L_2([0,1]^2)$$

$$f \in \mathcal{H} \subset L_2, \dim(\mathcal{H}) < \infty$$

Objective



Rectilinear transport in known scene

$$\xrightarrow{G}$$

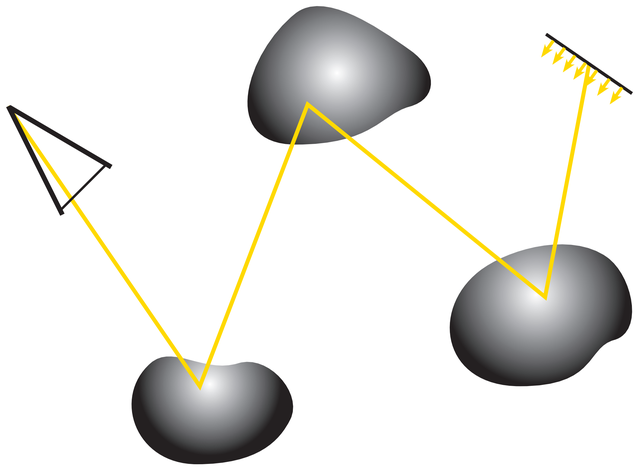
minimal # of rays for

$$\|\hat{f} - f\| < \epsilon$$

$$\hat{f} \in L_2([0,1]^2)$$

$$f \in \mathcal{H} \subset L_2, \dim(\mathcal{H}) < \infty$$

Objective



Rectilinear transport in
known scene

$$\xrightarrow{G}$$

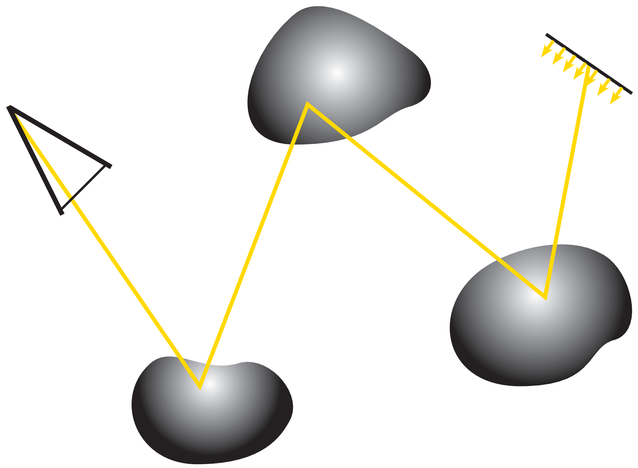
minimal # of rays for

$$\|\hat{f} - f\| \lesssim N^{-\alpha}$$

$$\hat{f} \in L_2([0,1]^2)$$

$$f \in \mathcal{H} \subset L_2, \dim(\mathcal{H}) < \infty$$

Objective



Rectilinear transport in known scene

$$\xrightarrow{G}$$

minimal # of rays for

$$\|\hat{f} - f\| \lesssim N^{-\alpha}$$

- Superresolution
- Progressive
- Varying resolution
- Parallelizable

$$\hat{f} \in L_2([0,1]^2)$$

$$f \in \mathcal{H} \subset L_2, \dim(\mathcal{H}) < \infty$$

Plan of attack

1. What's the cost of the last bounce?
2. But what has this to do with sampling?
3. And what about the function space?
4. How many rays do we need?
5. What has all of this to do with deep learning?

What's the cost of the last bounce?

$$C = N \cdot C_p$$

Number of pixels

Cost per pixel

Number of “pixels“

Number of “pixels“

compression

Number of “pixels”

compression

$\epsilon_{rel} = 6.44 \times 10^{-4}$

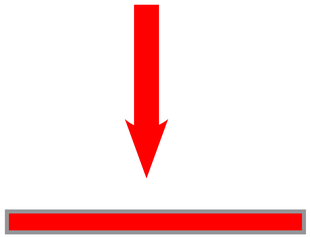
100%

8.48%

Number of “pixels“

compression

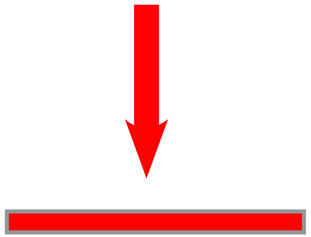
100%



Number of “pixels“

compression

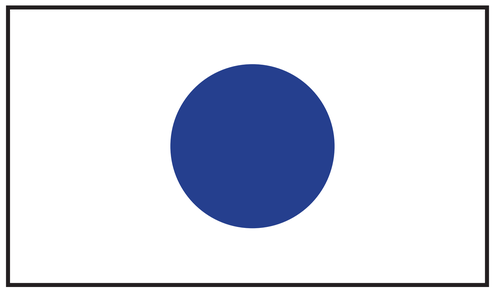
100%



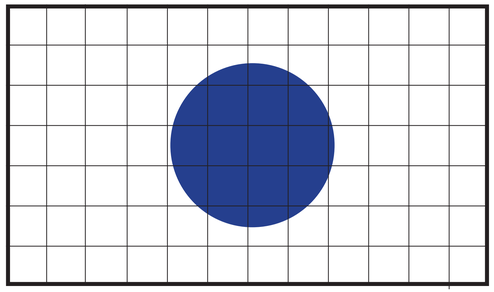
- wavelets (quasi)
- curvelets
- shearlets
- contourlets

$$\|\hat{f} - f\| \lesssim N^{-\alpha}$$

Number of “pixels“

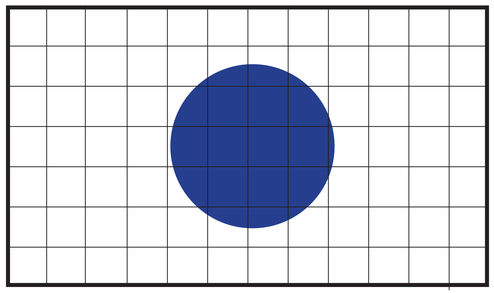


Number of “pixels“

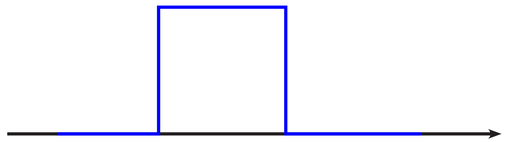


$$f(x) = \sum_{i \in U \subset r\mathbb{Z}^2} f_i \, \chi_i(x)$$

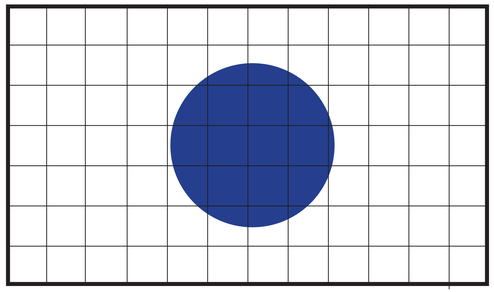
Number of “pixels“



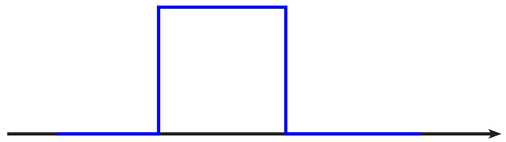
$$f(x) = \sum_{i \in U \subset r\mathbb{Z}^2} f_i \, \chi_i(x)$$

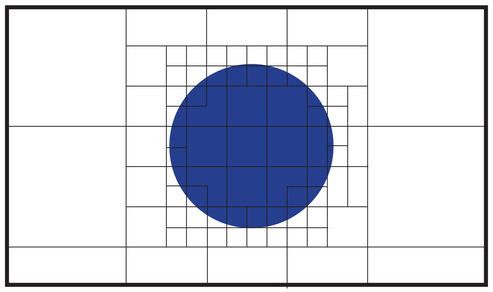


Number of “pixels“



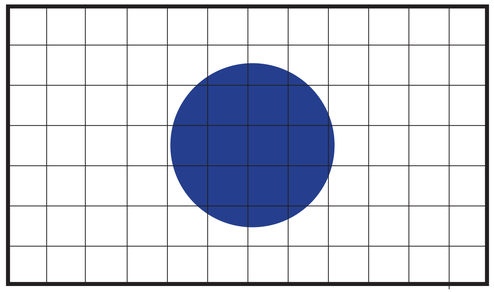
$$f(x) = \sum_{i \in U \subset r\mathbb{Z}^2} f_i \, \chi_i(x)$$



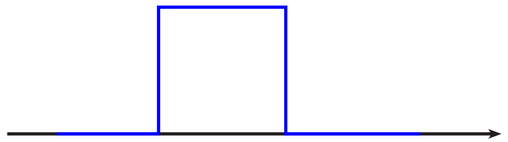


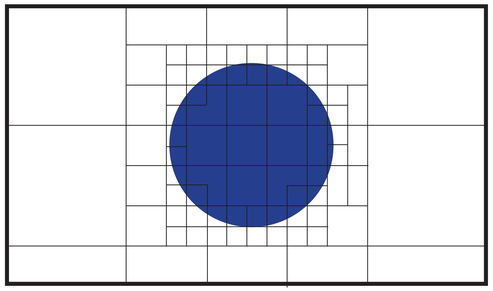
$$f(x) = \sum_{i \in \mathcal{I}} f_i \, \psi_i(x)$$

Number of “pixels“

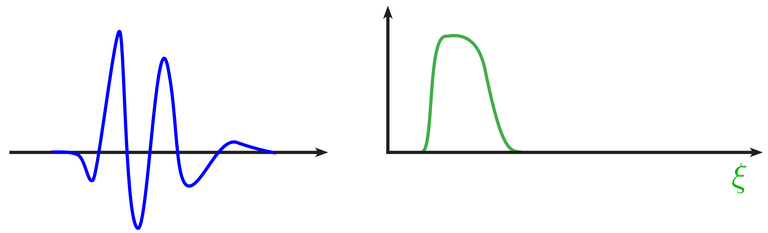


$$f(x) = \sum_{i \in U \subset r\mathbb{Z}^2} f_i \, \chi_i(x)$$

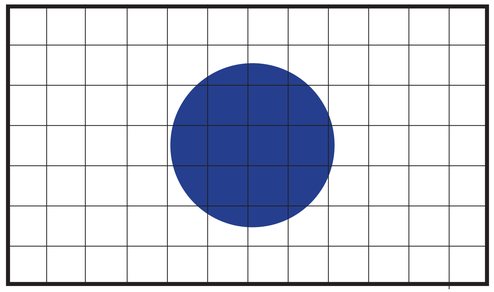




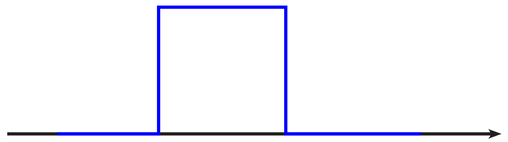
$$f(x) = \sum_{i \in \mathcal{I}} f_i \, \psi_i(x) \approx (2^{-l}k, 2^l \xi)$$

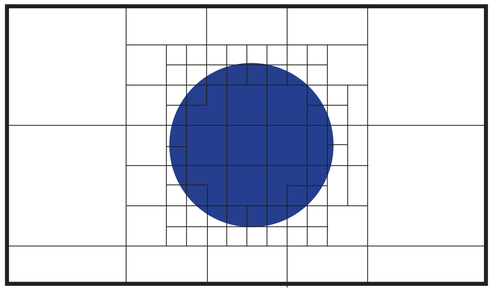


Number of “pixels“

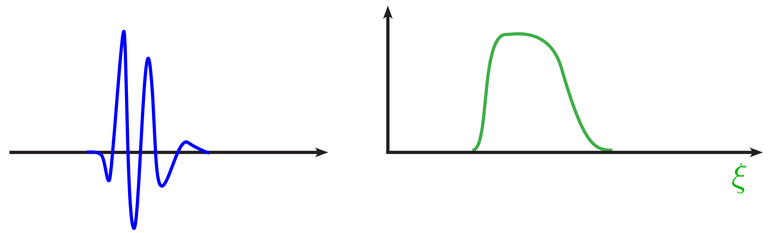


$$f(x) = \sum_{i \in U \subset r\mathbb{Z}^2} f_i \, \chi_i(x)$$

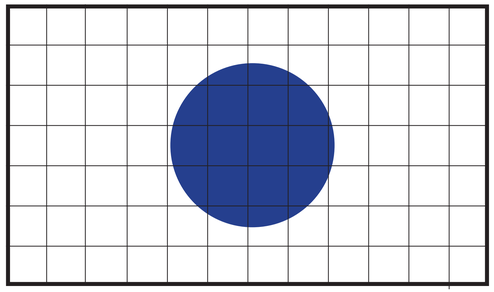




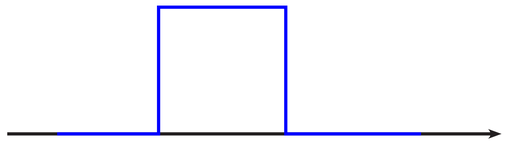
$$f(x) = \sum_{i \in \mathcal{I}} f_i \, \psi_i(x) \approx \left(2^{-l}k, 2^{l}\xi\right)$$

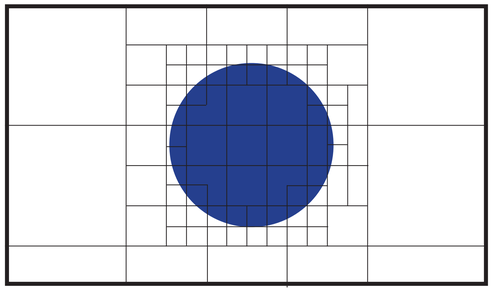


Number of “pixels”

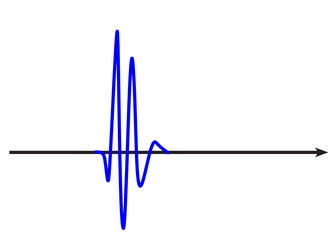


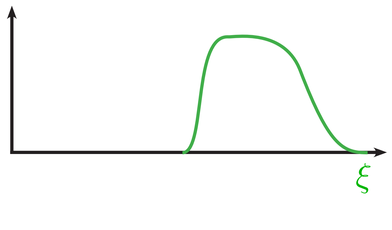
$$f(x) = \sum_{i \in U \subset r\mathbb{Z}^2} f_i \, \chi_i(x)$$



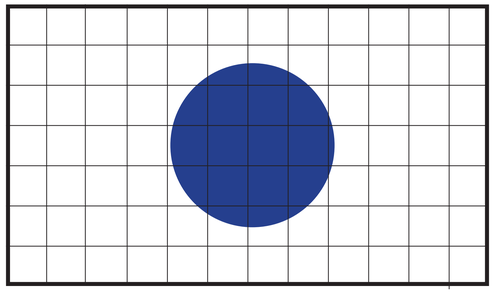


$$f(x) = \sum_{i \in \mathcal{I}} f_i \, \psi_i(x) \approx \left(2^{-l}k, 2^{l}\xi\right)$$

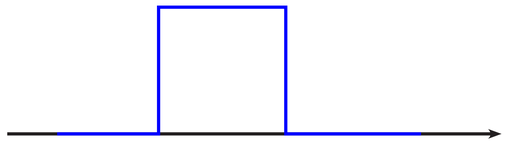


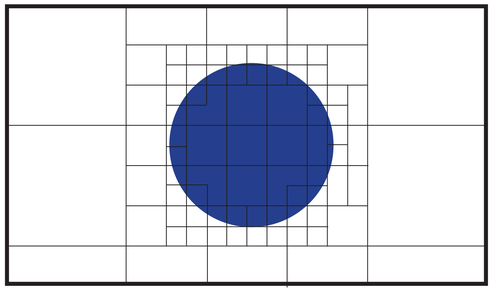


Number of “pixels“



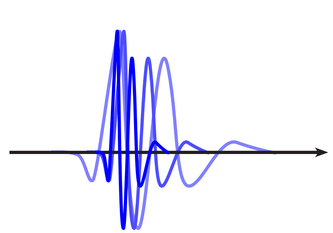
$$f(x) = \sum_{i \in U \subset r\mathbb{Z}^2} f_i \, \chi_i(x)$$

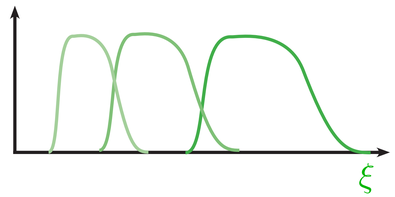




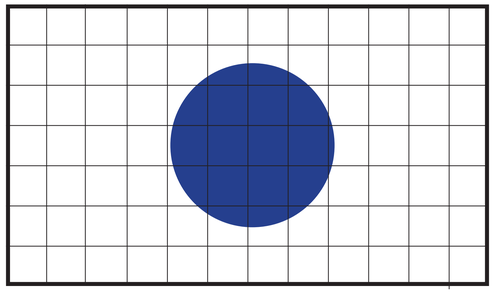
$$f(x) = \sum_{i \in \mathcal{I}} f_i \, \psi_i(x) \approx \left(2^{-l}k, 2^{l}\xi\right)$$

Hierarchical:
progressive

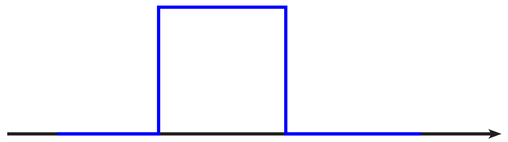


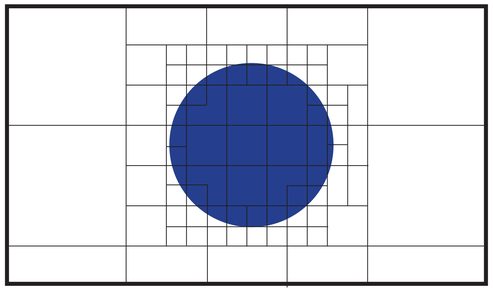


Number of “pixels”



$$f(x) = \sum_{i \in U \subset r\mathbb{Z}^2} f_i \, \chi_i(x)$$

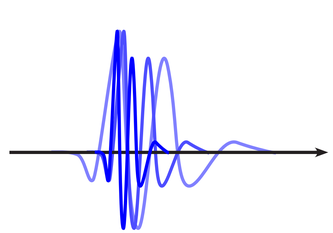


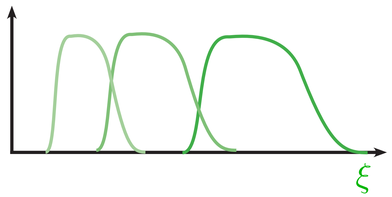


Locality: varying resolution

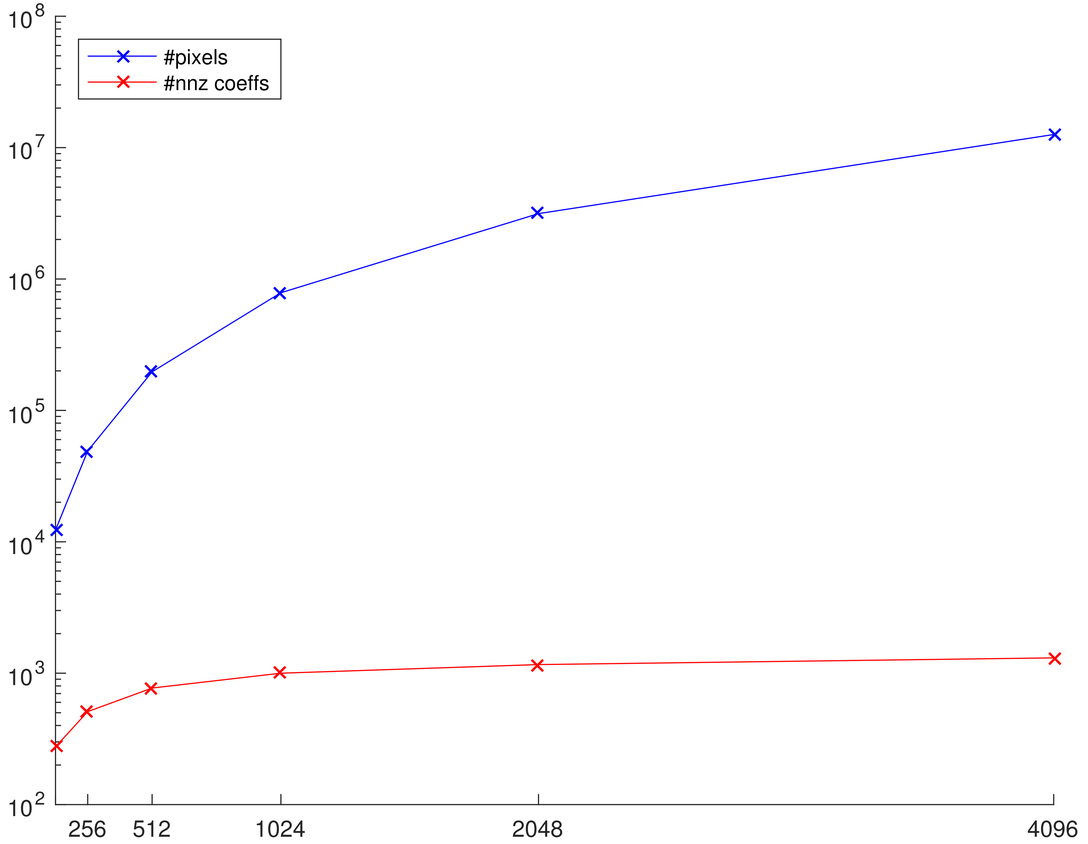
$$f(x) = \sum_{i \in \mathcal{I}} f_i \, \psi_i(x) \approx (2^{-l}k, 2^l \xi)$$

Hierarchical:
progressive

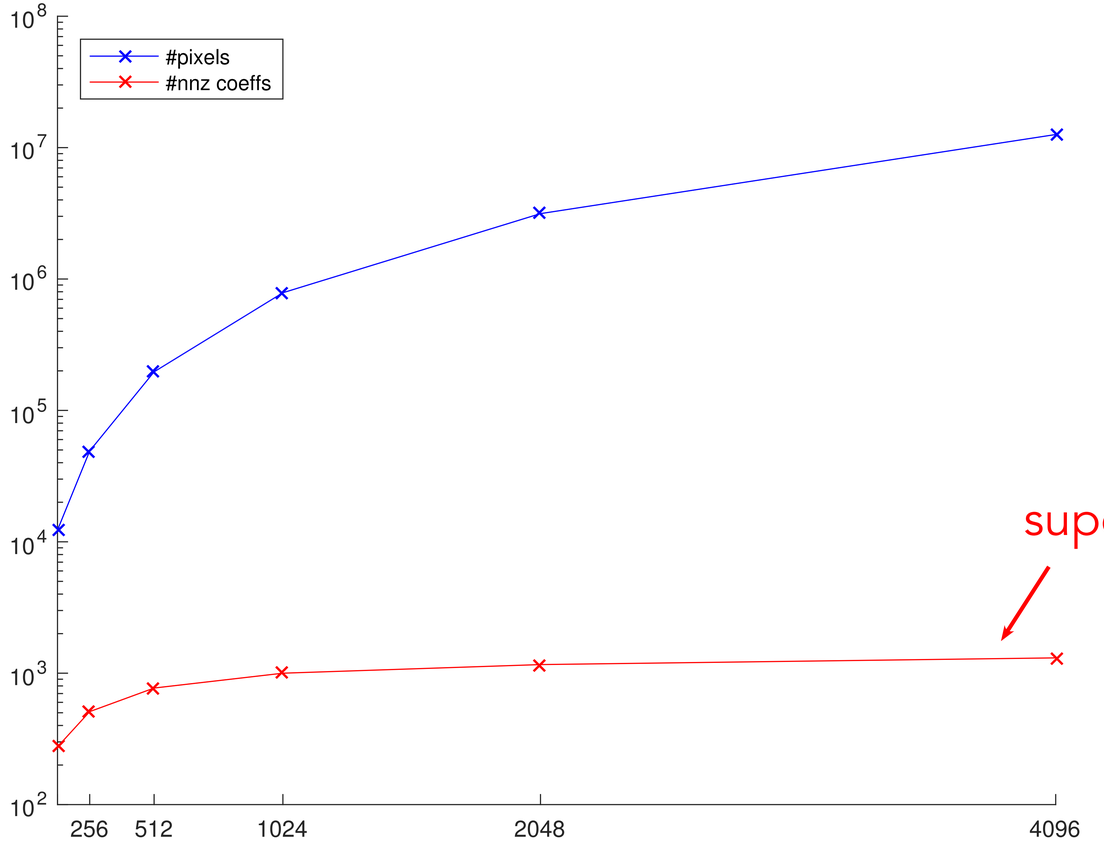




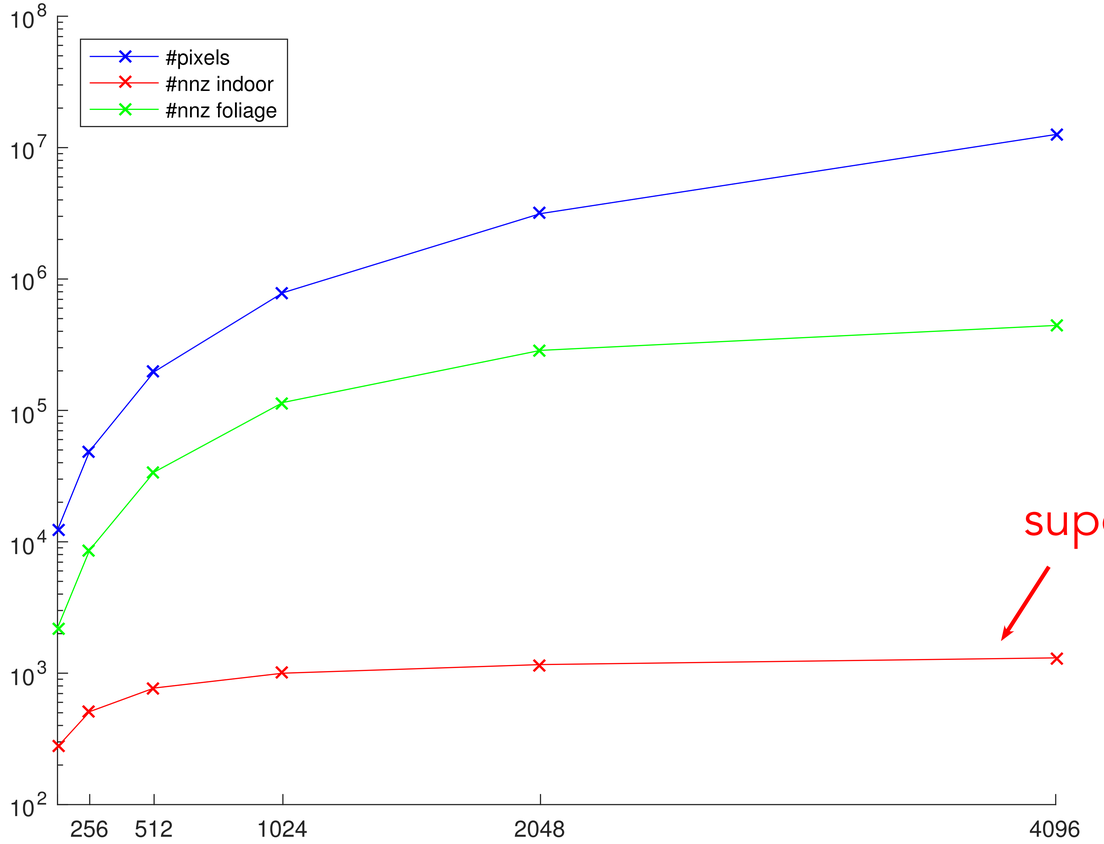
Number of “pixels”



Number of “pixels“

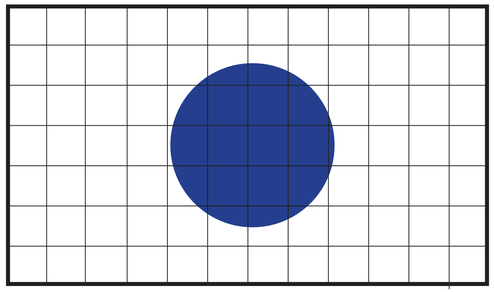


Number of “pixels”

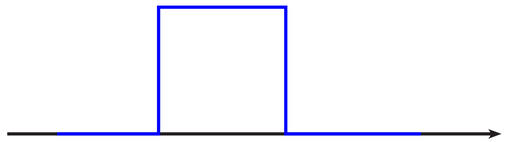


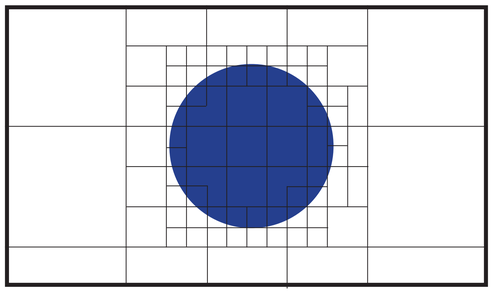
super-resolution

Number of “pixels“

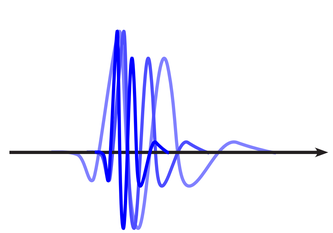


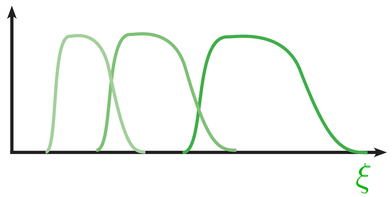
$$f(x) = \sum_{i \in U \subset r\mathbb{Z}^2} f_i \, \chi_i(x)$$



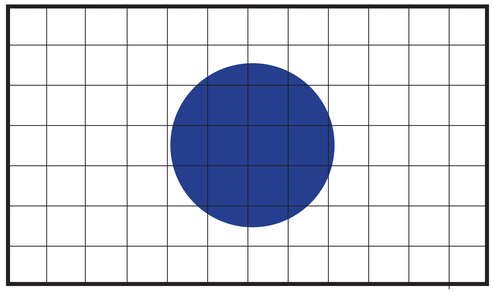


$$f(x) = \sum_{i \in \mathcal{I}} f_i \, \psi_i(x) \approx \left(2^{-l}k, 2^{l}\xi\right)$$

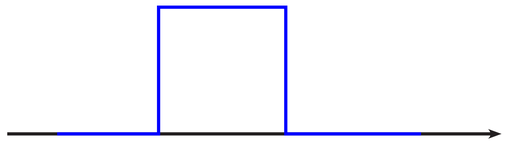


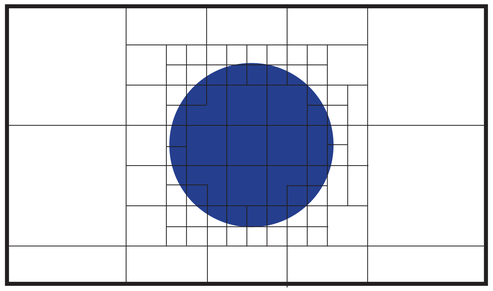


Number of “pixels“

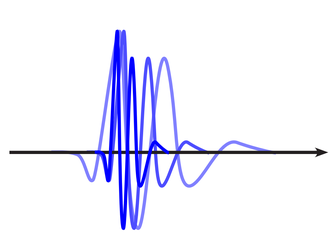


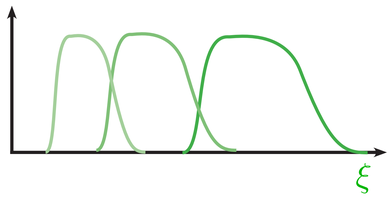
$$f(x) = \sum_{i \in U \subset r\mathbb{Z}^2} f_i \, \chi_i(x)$$



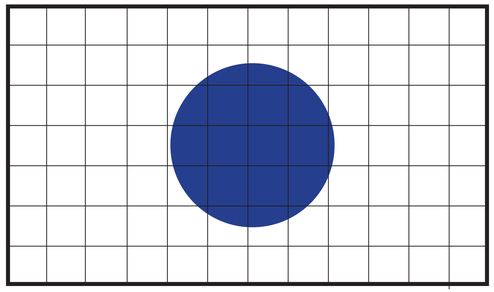


$$f(x) = \sum_{i \in \mathcal{I}} f_i \, \psi_i(x) \approx (2^{-l}k, 2^l \xi)$$

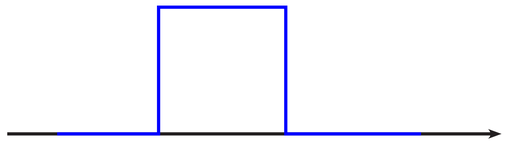


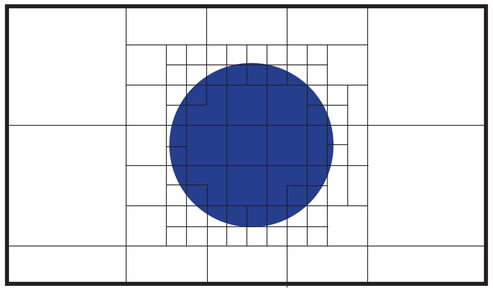


Number of “pixels”



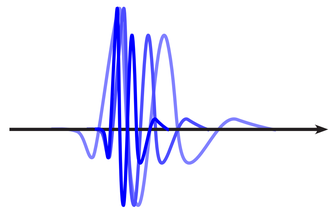
$$f(x) = \sum_{i \in U \subset r\mathbb{Z}^2} f_i \, \chi_i(x)$$

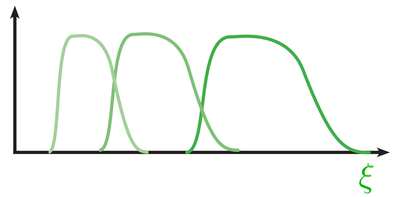




$$f(x) = \sum_{i \in \mathcal{I}} f_i \, \psi_i(x) \approx \left(2^{-l}k, 2^{l}\xi\right)$$

depends on image function





Computational costs

$$C = N \cdot C_p$$

Number of "pixels"

Cost per pixel

Computational costs

$$C = N \cdot C_p$$

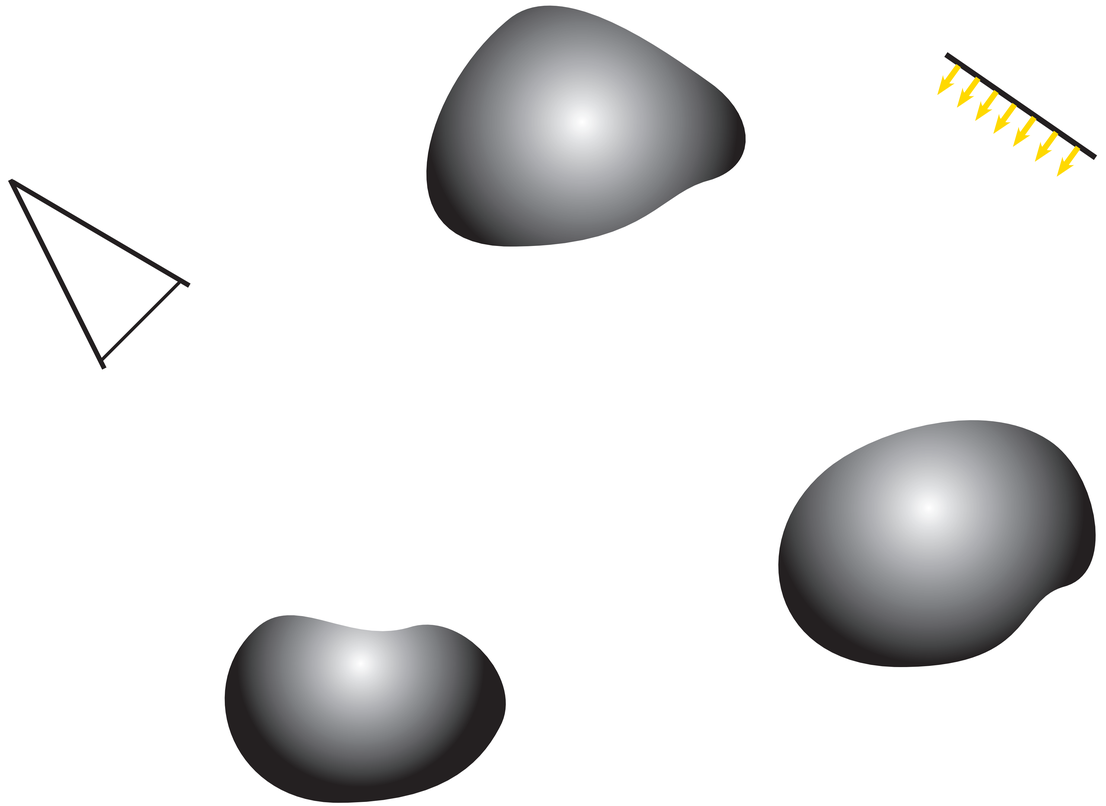
Number of "pixels"

Cost per pixel

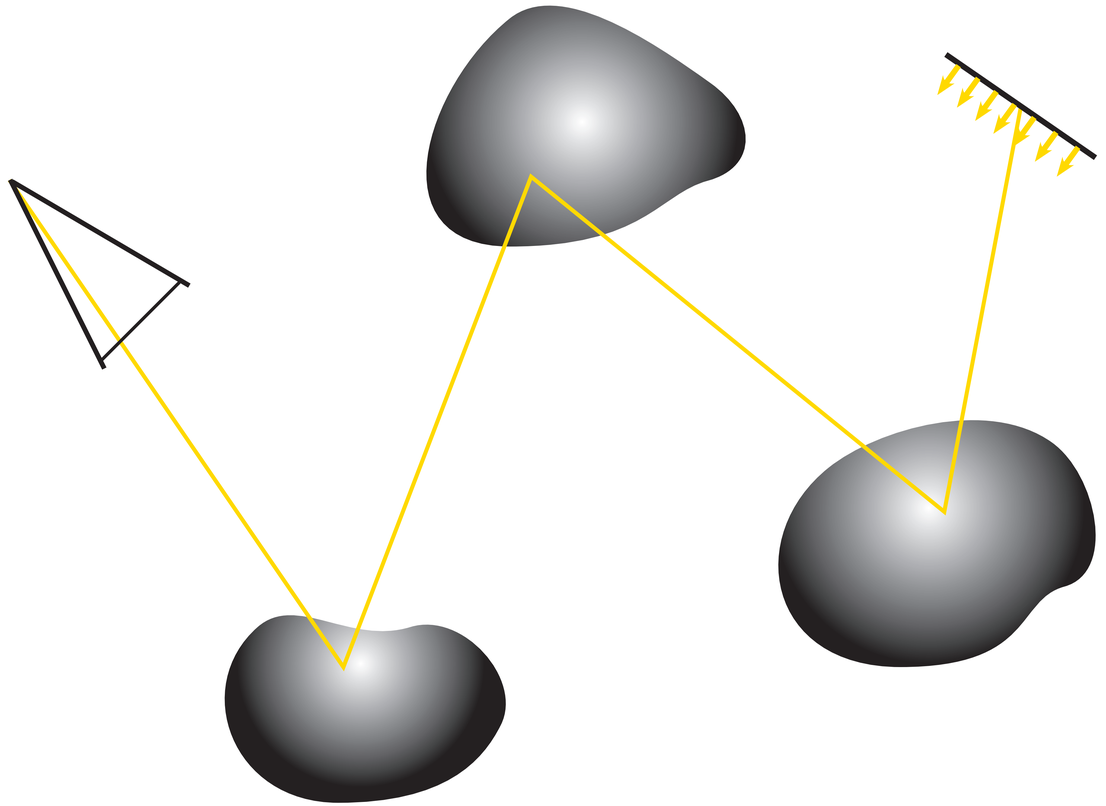
minimize by

- Use sparse, adaptive image representation

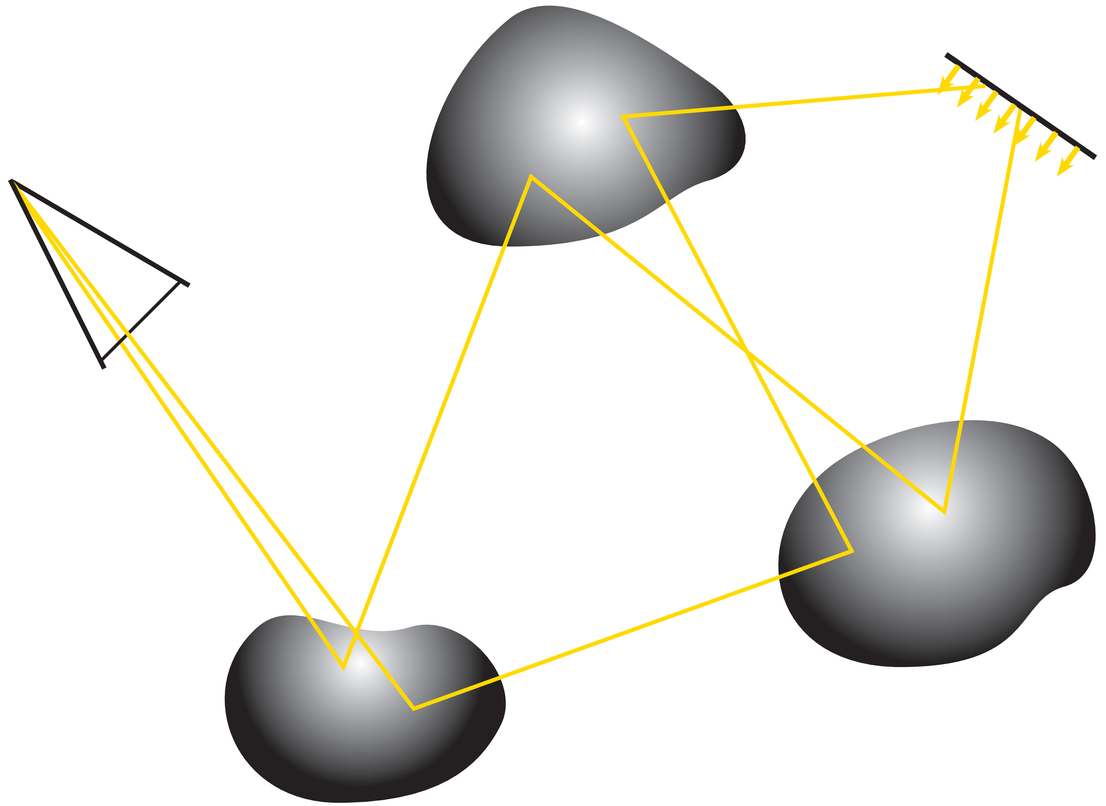
Cost per “pixel”



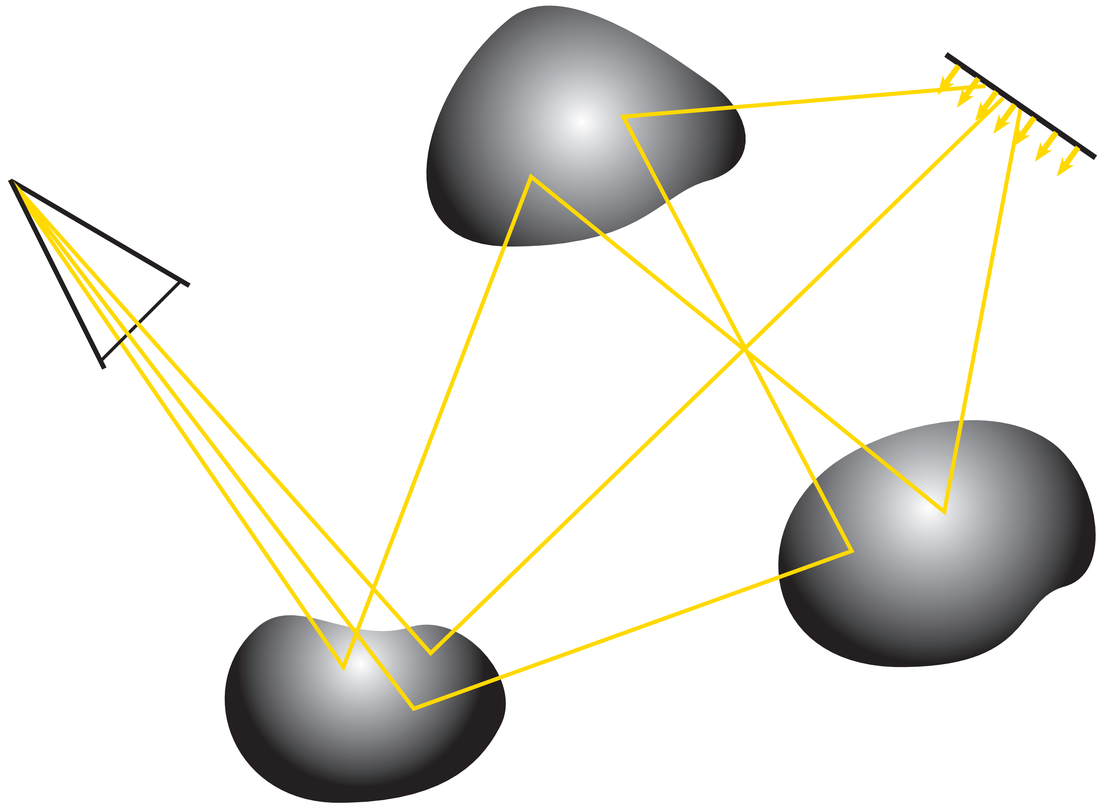
Cost per “pixel”



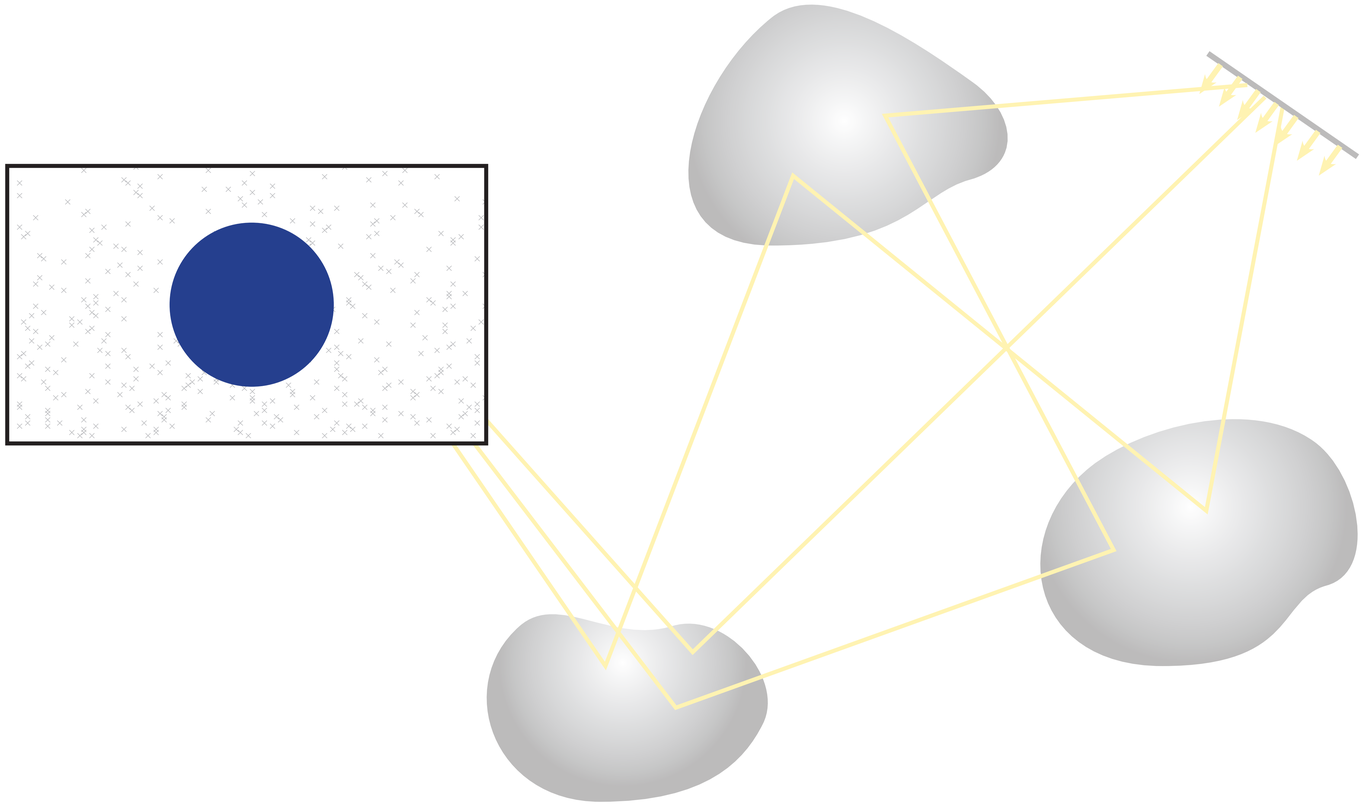
Cost per “pixel”



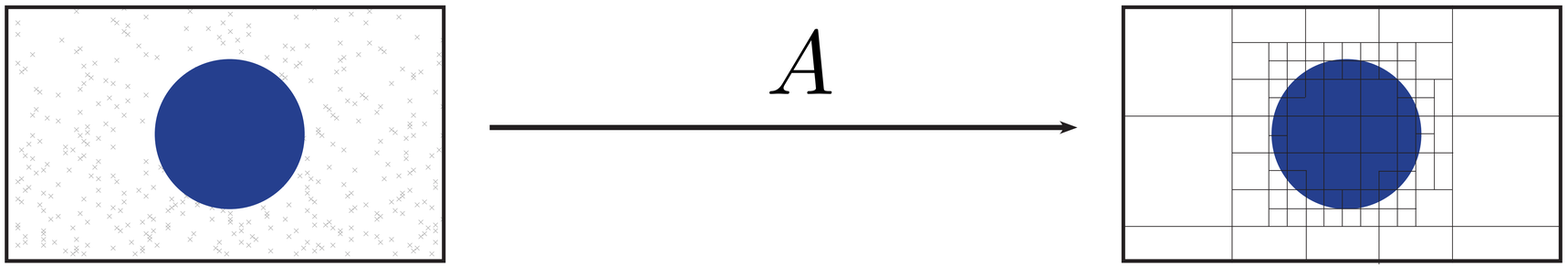
Cost per “pixel”



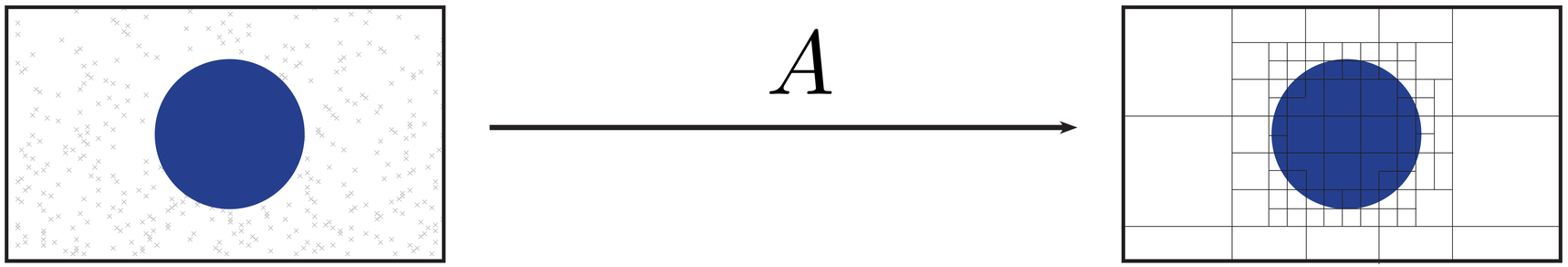
Cost per “pixel”



Cost per “pixel”



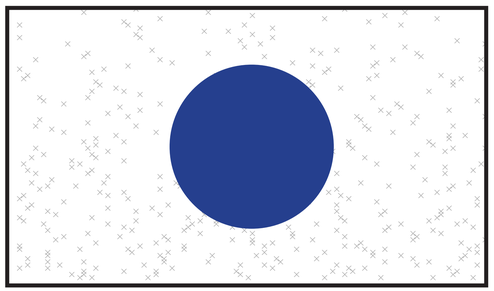
Cost per “pixel”



$$\left\{ f(x_i) \right\} \in \mathbb{R}^m$$

$$\left\{ f_i \right\} \in \mathbb{R}^n$$

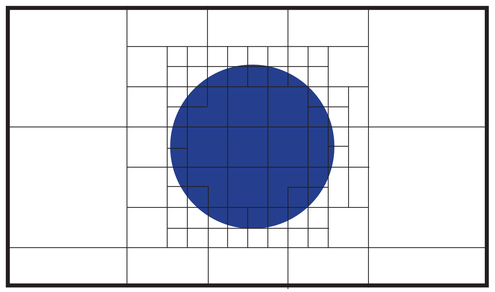
Cost per “pixel”



$$\left\{ f(x_i) \right\} \in \mathbb{R}^m$$

$$A$$

$$f(x_j) = \sum_{i \in \mathcal{I}} f_i \, \phi_i(x_j)$$



$$\left\{ f_i \right\} \in \mathbb{R}^n$$

Cost per “pixel”

$$f(x_j) = \sum_{i \in \mathcal{I}} f_i \, \phi_i(x_j)$$

Cost per “pixel”

$$f(x_j) = \sum_{i \in \mathcal{I}} f_i \, \phi_i(x_j)$$

$$\Updownarrow$$

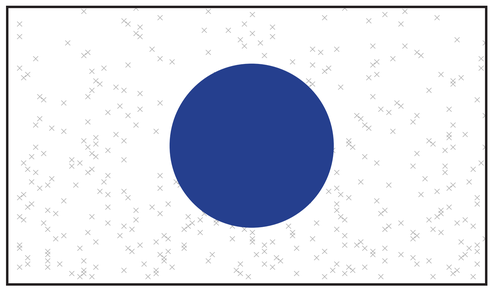
$$\begin{pmatrix} f(x_1) \\ \vdots \\ f(x_m) \end{pmatrix} = \begin{pmatrix} \phi_1(x_1) & \phi_2(x_1) & \dots & & \\ & \phi_2(x_2) & \phi_3(x_1) & \dots & \\ & & \ddots & \ddots & \\ & & \dots & \phi_{n-1}(x_m) & \phi_n(x_m) \end{pmatrix} \begin{pmatrix} f_1 \\ \vdots \\ f_n \end{pmatrix}$$

Cost per “pixel”

$$f(x_j) = \sum_{i \in \mathcal{I}} f_i \, \phi_i(x_j)$$

$$\underbrace{\begin{pmatrix} f(x_1) \\ \vdots \\ f(x_m) \end{pmatrix}}_{\bar{f}(x_i)} = \begin{pmatrix} \phi_1(x_1) & \phi_2(x_1) & \dots & & \\ & \phi_2(x_2) & \phi_3(x_1) & \dots & \\ & & \ddots & \ddots & \\ & & \dots & \phi_{n-1}(x_m) & \phi_n(x_m) \end{pmatrix} \underbrace{\begin{pmatrix} f_1 \\ \vdots \\ f_n \end{pmatrix}}_{\bar{f}_i}$$

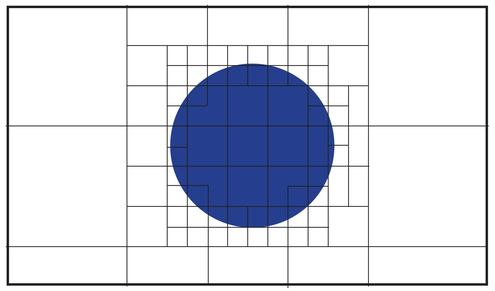
Cost per “pixel”



$$\left\{ f(x_i) \right\} \in \mathbb{R}^m$$

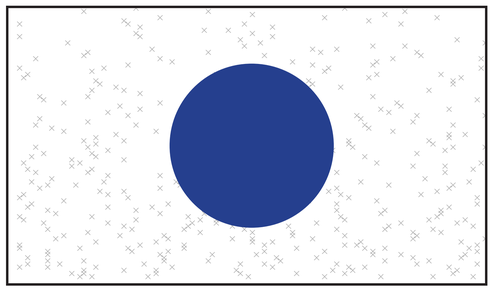
$$A$$

$$\bar{f}(x_i) = B\, \bar{f}_i$$



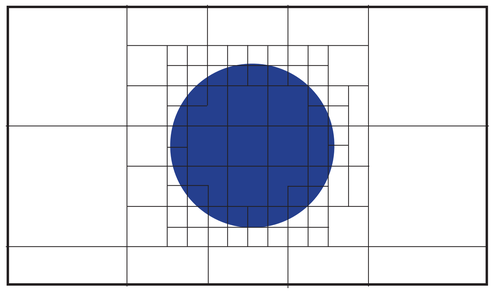
$$\left\{ f_i \right\} \in \mathbb{R}^n$$

Cost per “pixel”



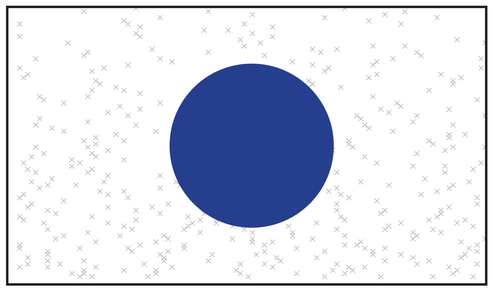
$$\bar{f}_i = B^+ \bar{f}(x_i)$$

$$\bar{f}(x_i) = B\, \bar{f}_i$$



$$\left\{ f(x_i) \right\} \in \mathbb{R}^m \qquad \left\{ f_i \right\} \in \mathbb{R}^n$$

Cost per “pixel”

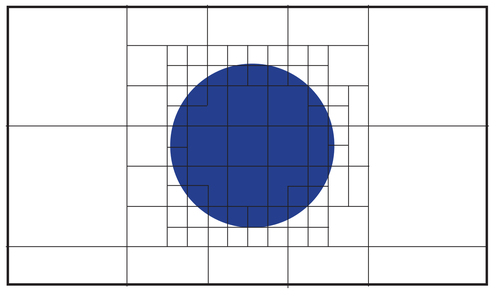


$$\left\{ f(x_i) \right\} \in \mathbb{R}^m$$

pseudo inverse

$$\bar{f}_i = B^+ \bar{f}(x_i)$$

$$\bar{f}(x_i) = B\, \bar{f}_i$$



$$\left\{ f_i \right\} \in \mathbb{R}^n$$

Cost per “pixel”

Example: pixel basis

$$B = \begin{pmatrix} \phi_1(x_1) & \phi_2(x_1) & \dots & & \\ & \phi_2(x_2) & \phi_3(x_1) & \dots & \\ & & \ddots & \ddots & \\ & & \dots & \phi_{n-1}(x_m) & \phi_n(x_m) \end{pmatrix}$$

Cost per “pixel”

Example: pixel basis

$$B = \begin{pmatrix} \chi_1(x_1) & \chi_2(x_1) & \cdots & & \\ & \chi_2(x_2) & \chi_3(x_1) & \cdots & \\ & & \ddots & \ddots & \\ & & \cdots & \chi_{n-1}(x_m) & \chi_n(x_m) \end{pmatrix}$$

Cost per “pixel”

Example: pixel basis

$$B = \begin{pmatrix} 1 & 0 & \dots & & & \\ 1 & 0 & \dots & & & \\ & & & \ddots & \ddots & \\ & & & \dots & 0 & 1 \\ & & & \dots & 0 & 1 \end{pmatrix}$$

Cost per “pixel”

Example: pixel basis

$$B = \begin{pmatrix} \blacksquare & 0 & \cdots & & \\ 0 & \blacksquare & & & \\ & & \ddots & \ddots & \\ & & \cdots & 0 & \blacksquare \\ & & \cdots & 0 & \end{pmatrix}$$

Cost per “pixel”

Example: pixel basis

$$B^+ = \begin{pmatrix} \blacksquare & \cdots & & \\ & \blacksquare & & \\ & & \ddots & \ddots \\ & & & \blacksquare \end{pmatrix}$$

Cost per “pixel”

Example: pixel basis

$$B^+ = \begin{pmatrix} 1/N_1 & 1/N_1 & \dots & & \\ & & 1/N_2 & 1/N_2 & \\ & & & \ddots & \ddots \\ & & & & 1/N_n & 1/N_n \end{pmatrix}$$

Cost per “pixel”

Example: pixel basis

$$B^+ = \begin{pmatrix} 1/N_1 & 1/N_1 & \dots & & \\ & & 1/N_2 & 1/N_2 & \\ & & & \ddots & \ddots \\ & & & & 1/N_n & 1/N_n \end{pmatrix}$$

$$\Longrightarrow \quad f_i = \frac{1}{N_i} \sum_{j=1}^{N_i} f(x_j)$$

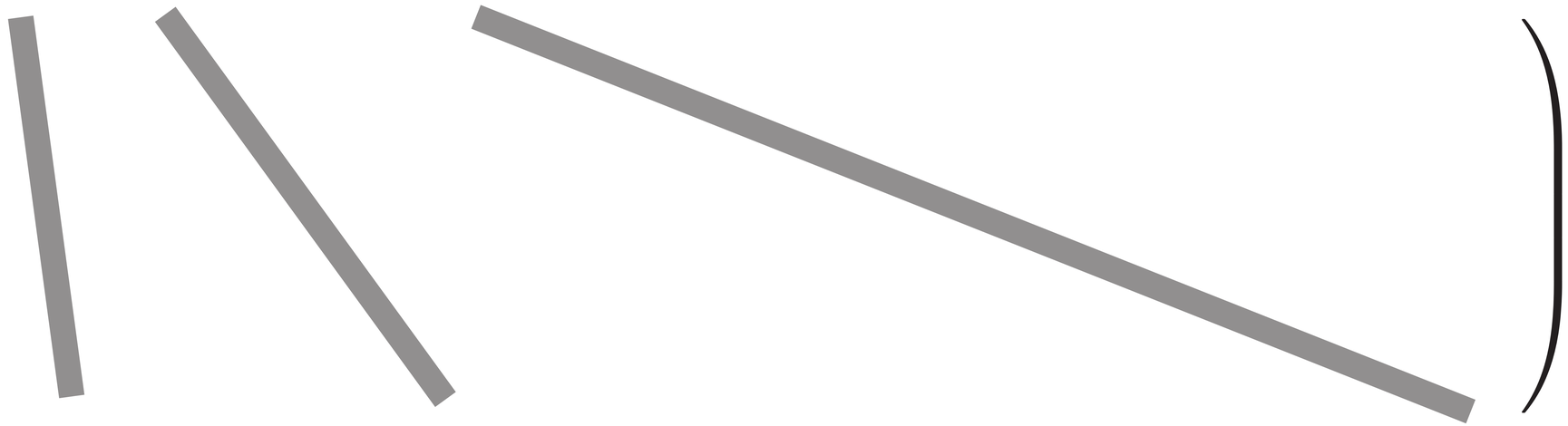
Cost per “pixel”

Example: wavelet basis

$$B = \begin{pmatrix} \psi_1(x_1) & \psi_2(x_1) & \dots & & \\ & \psi_2(x_2) & \psi_3(x_1) & \dots & \\ & & \ddots & \ddots & \\ & & \dots & \psi_{n-1}(x_m) & \psi_n(x_m) \end{pmatrix}$$

Cost per “pixel”

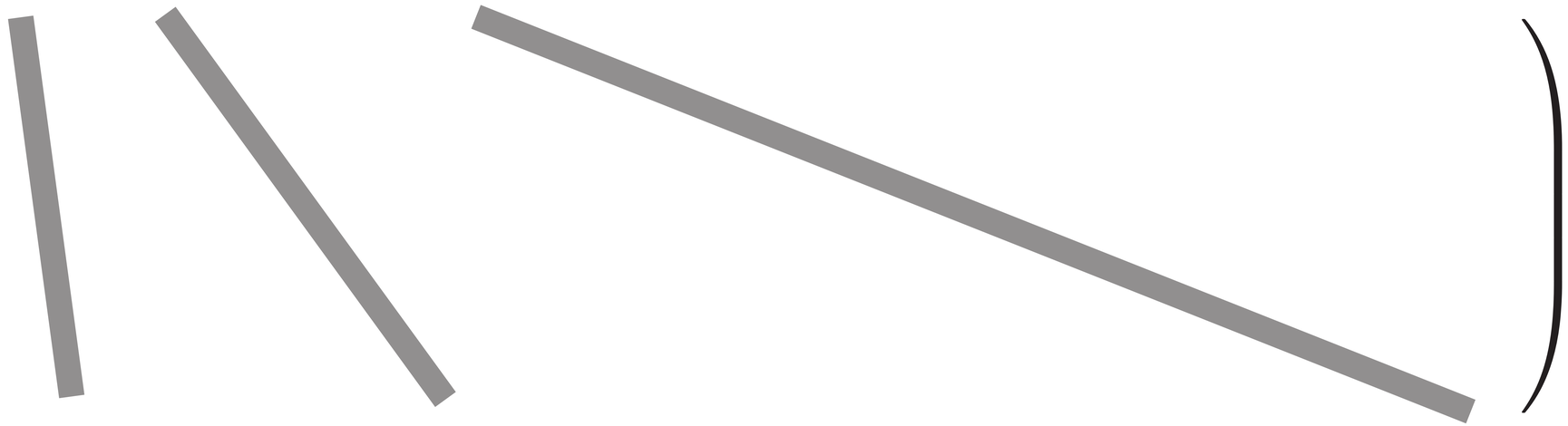
Example: wavelet basis

$$B = \left(\quad \right)$$


Cost per “pixel”

Example: wavelet basis

$$B = \left(\quad \right)$$

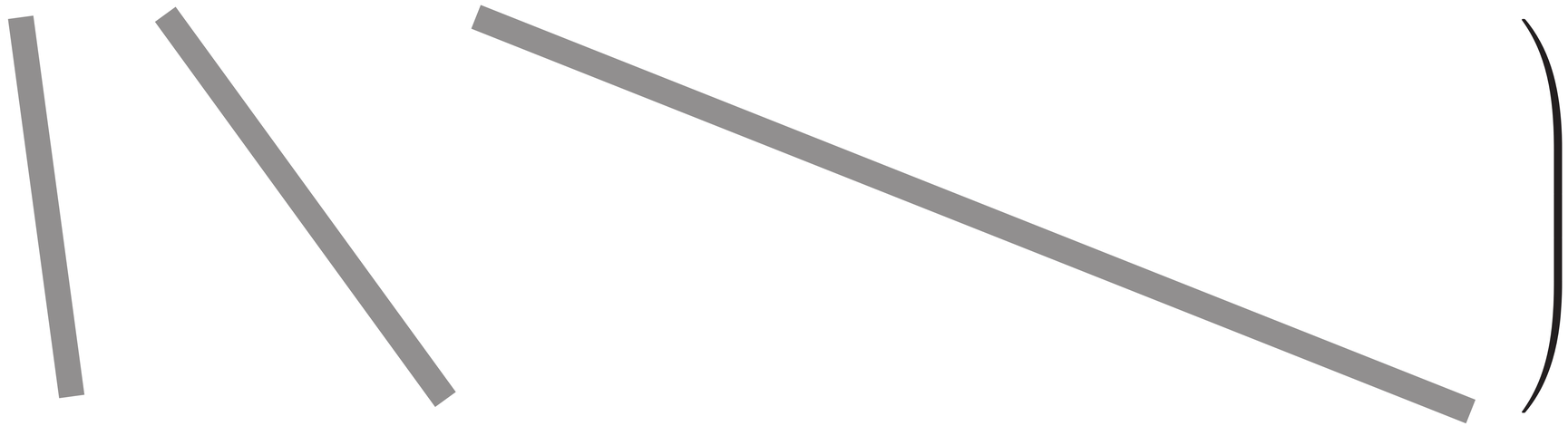


$$B^+ \;\not\cong\; f_i = \frac{1}{N} \sum_{i=1}^{N} f(x_i)\, \psi(x_i)$$

Cost per “pixel”

Example: wavelet basis

$$B = \left(\quad \right)$$

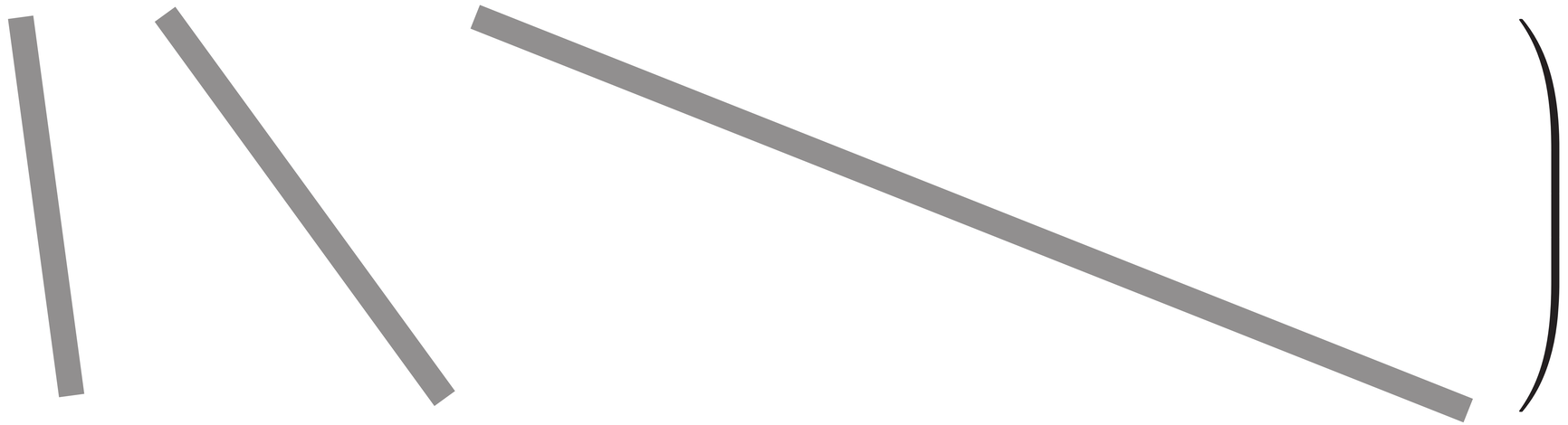


Multi-res structure: can be solved in O(n) time using multi-grid.

Cost per “pixel”

Example: wavelet basis

$$B = \left(\quad \right)$$



Multi-res structure: can be solved in O(n) time using multi-grid.

optimal

What's the cost of the last bounce?

$$C = N \cdot C_p$$

Number of "pixels"

Cost per pixel

minimize by

- Use sparse, adaptive image representation

What's the cost of the last bounce?

$$C = N \cdot C_p$$

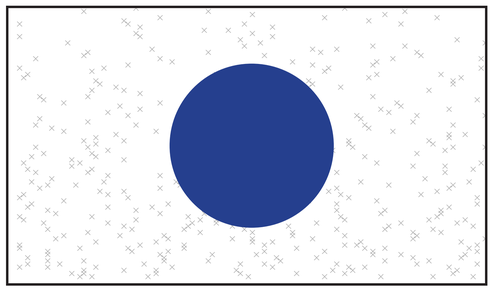
Number of "pixels"

Cost per pixel

minimize by

- Use sparse, adaptive image representation
- Use at least *n* samples
- Solve using multi-grid

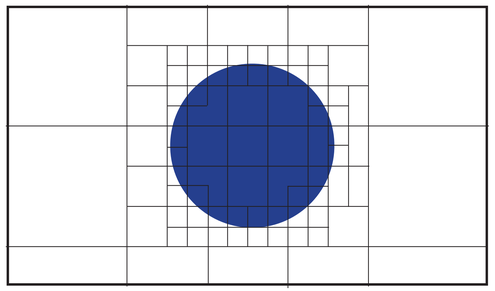
But what has this to do with sampling?



$$\left\{ f(x_i) \right\} \in \mathbb{R}^m$$

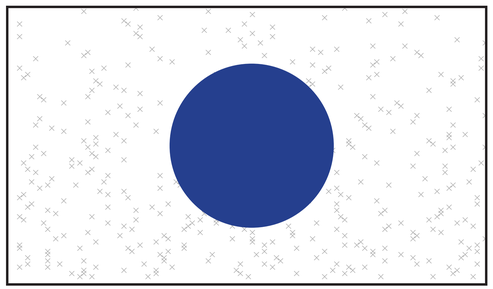
$$\bar{f}_i = B^+ \bar{f}(x_i)$$

$$\bar{f}(x_i) = B\, \bar{f}_i$$



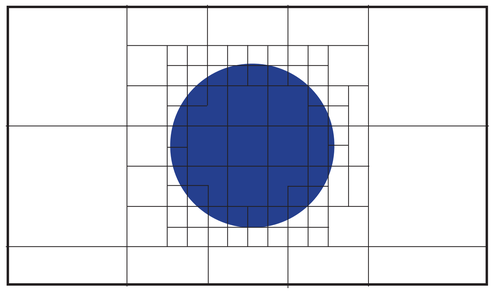
$$\left\{ f_i \right\} \in \mathbb{R}^n$$

But what has this to do with sampling?



$$\bar{f}_i = B^+ \bar{f}(x_i)$$

$$\bar{f}(x_i) = B\, \bar{f}_i$$



$$\left\{ f(x_i) \right\} \in \mathbb{R}^m$$

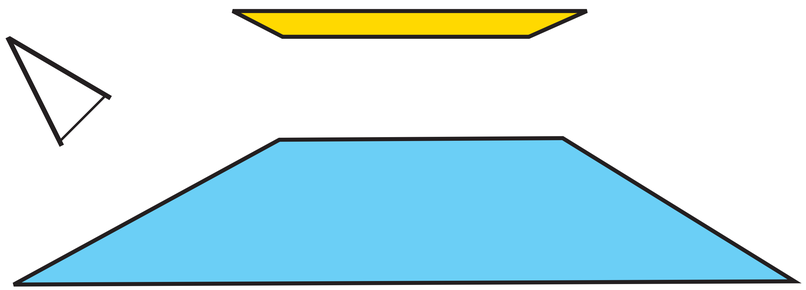
$$\left\{ f_i \right\} \in \mathbb{R}^n$$

But what has this to do with sampling?

$$\left\{ f(x_i) \right\} = \left\{ \delta_{x_i}(f) \right\}$$

But what has this to do with sampling?

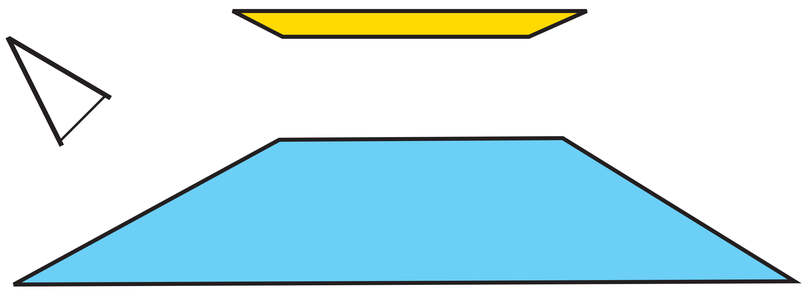
$$\left\{ f(x_i) \right\} = \left\{ \delta_{x_i}(f) \right\}$$



$$f(x) = L_{\text{out}}(x)$$

But what has this to do with sampling?

$$\underbrace{\left\{ f(x_i) \right\}}_{f = \hat{f} + \epsilon} = \left\{ \delta_{x_i}(f) \right\}$$

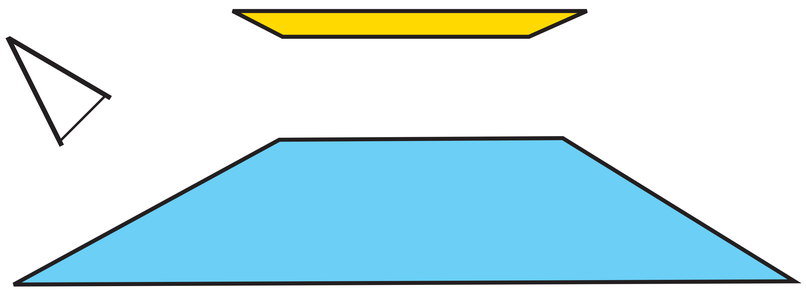


$$f(x) = L_{\text{out}}(x)$$

But what has this to do with sampling?

$$\underbrace{\left\{ f(x_i) \right\}} = \left\{ \delta_{x_i}(f) \right\}$$

$$f = \hat{f} + \epsilon$$

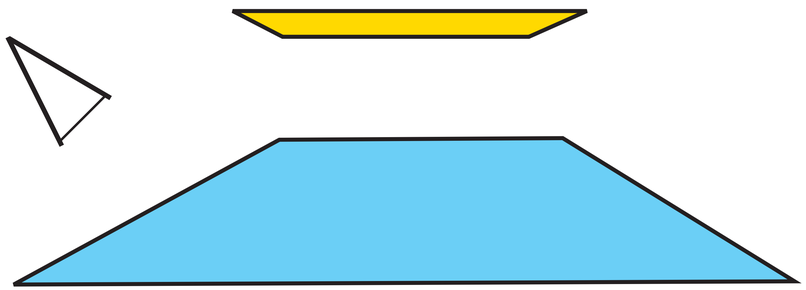


$$f(x) = L_{\text{out}}(x)$$

$$\Longrightarrow \quad \delta_{x_i}(f) \approx \delta_{x_i}(\hat{f})$$

But what has this to do with sampling?

$$\underbrace{\left\{ f(x_i) \right\}}_{f = \hat{f} + \epsilon} = \left\{ \delta_{x_i}(f) \right\}$$



$$f(x) = L_{\text{out}}(x)$$

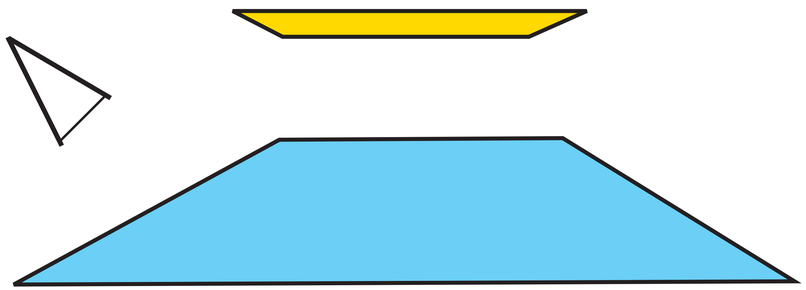
$$\Longrightarrow \boxed{\delta_{x_i}(f) \approx \delta_{x_i}(\hat{f})}$$

δ_x is a continuous functional

But what has this to do with sampling?

$$\underbrace{\left\{ f(x_i) \right\}} = \left\{ \delta_{x_i}(f) \right\}$$

$$f = \hat{f} + \epsilon$$



$$f(x) = L_{\text{out}}(x)$$

$$\Longrightarrow \boxed{\delta_{x_i}(f) \approx \delta_{x_i}(\hat{f})}$$

δ_x is a continuous functional

$$\Longrightarrow f \in \mathcal{H}$$ with $\mathcal{H}$ a "reasonable" function space

Reproducing kernel Hilbert spaces

$\mathcal{H}(X)$ is Hilbert and $\delta_{x_i}(f)$ continuous, then

Reproducing kernel Hilbert spaces

$\mathcal{H}(X)$ is Hilbert and $\delta_{x_i}(f)$ continuous, then

$$\forall \bar{x} \in X, \exists k_{\bar{x}}(x) \in \mathcal{H}(X) \;\; : \;\; \left\langle k_{\bar{x}}(x) \, , \; f(x) \right\rangle = f(\bar{x})$$

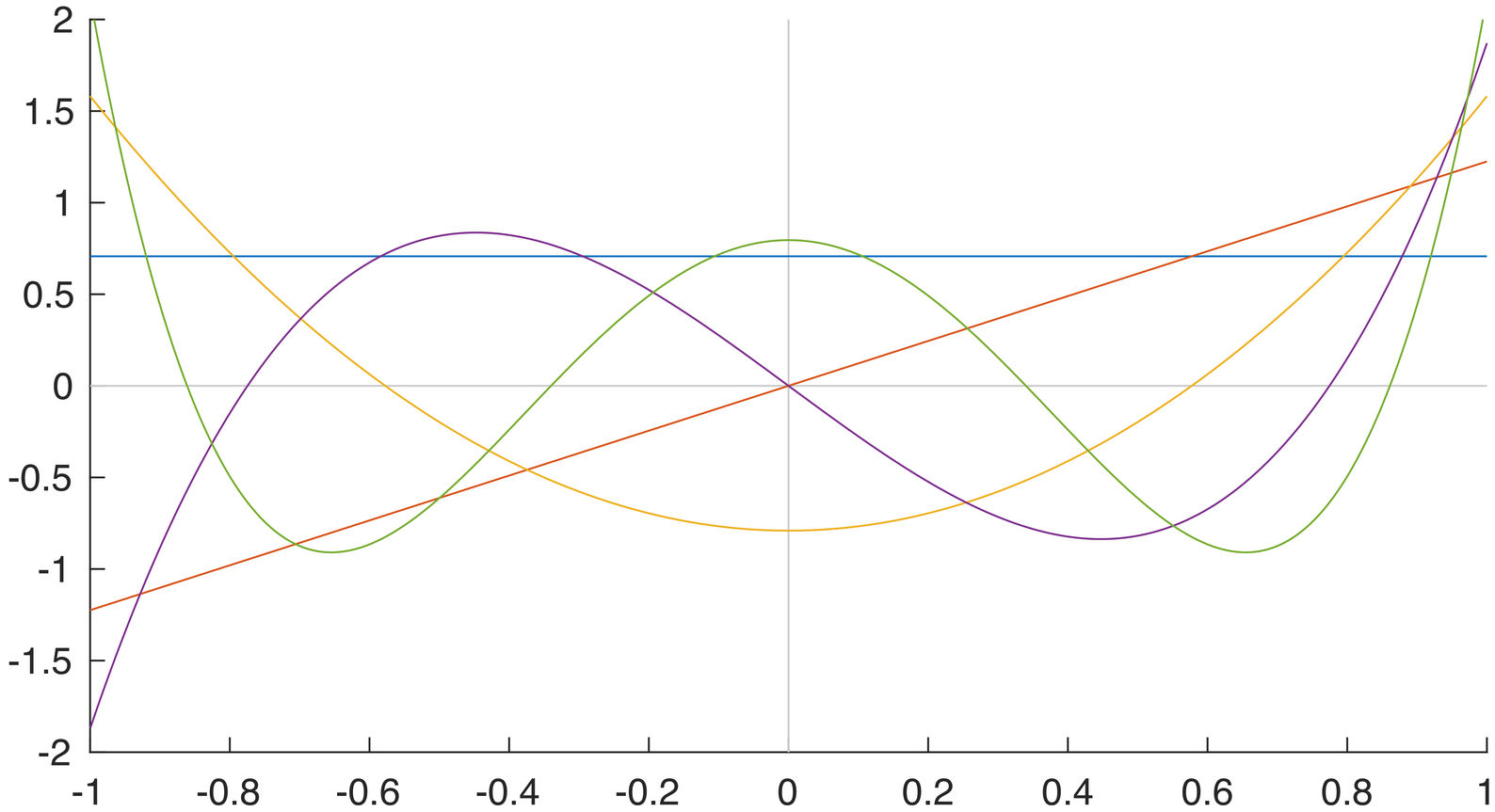
Reproducing kernel Hilbert spaces

$\mathcal{H}(X)$ is Hilbert and $\delta_{x_i}(f)$ continuous, then

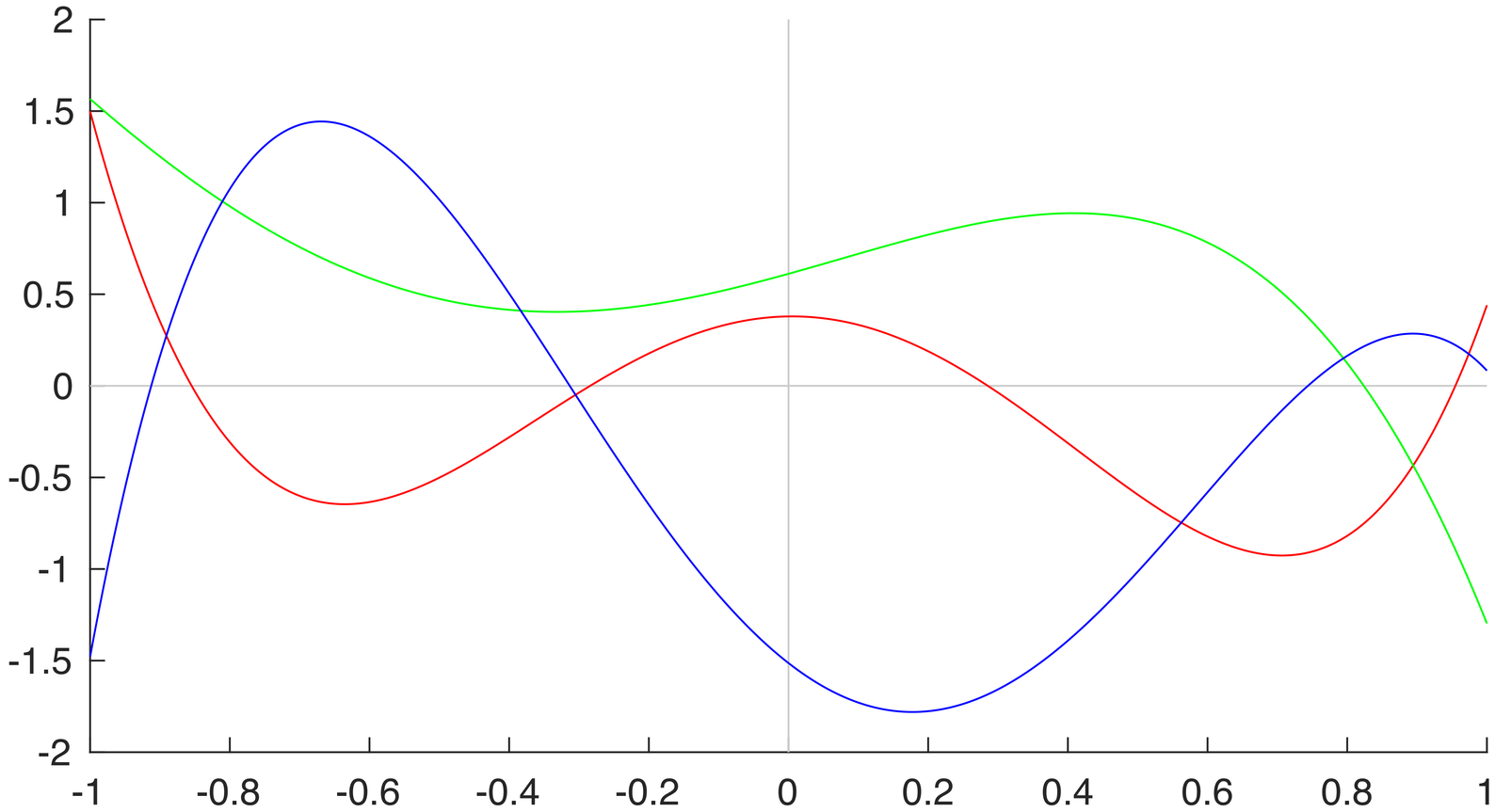
$$\forall \bar{x} \in X, \exists k_{\bar{x}}(x) \in \mathcal{H}(X) \;\; : \;\; \left\langle k_{\bar{x}}(x), \; f(x) \right\rangle = f(\bar{x})$$

reproducing kernel

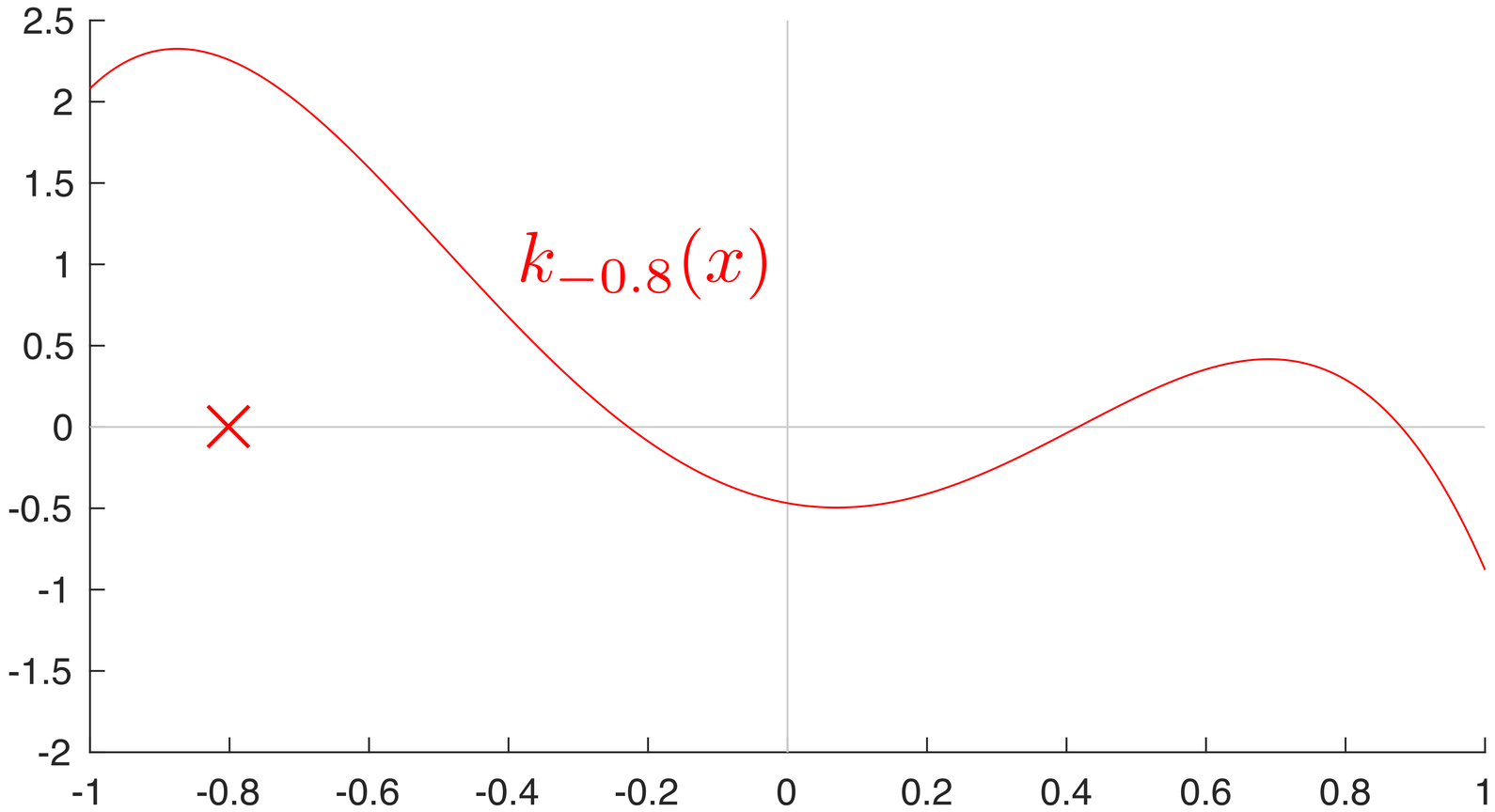
Example: $\mathcal{H}([-1,1]) = \mathrm{span}_{n=1\cdots N}\{P_n(x)\}$



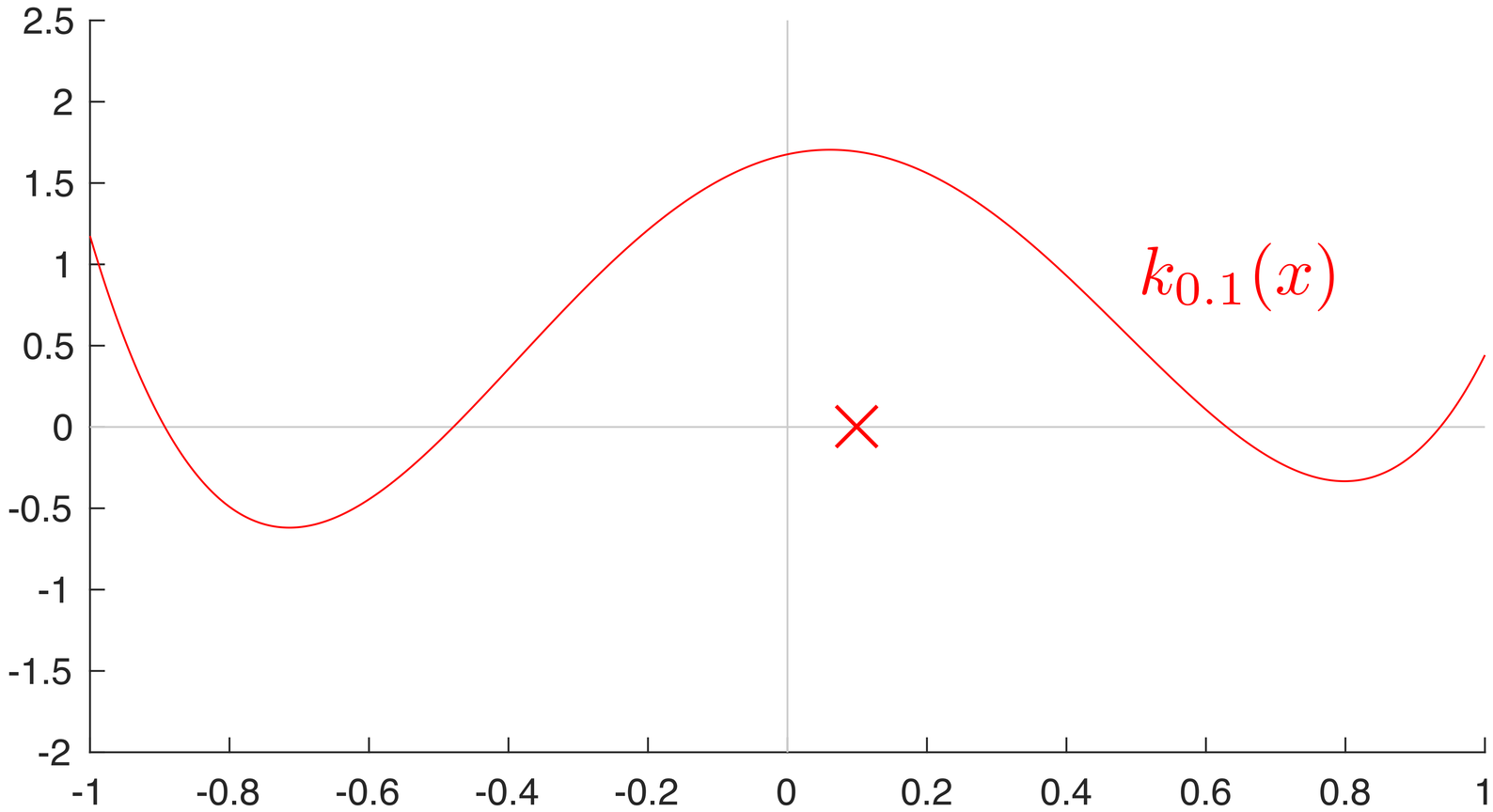
Example: $\mathcal{H}([-1,1]) = \mathrm{span}_{n=1\cdots N}\{P_n(x)\}$



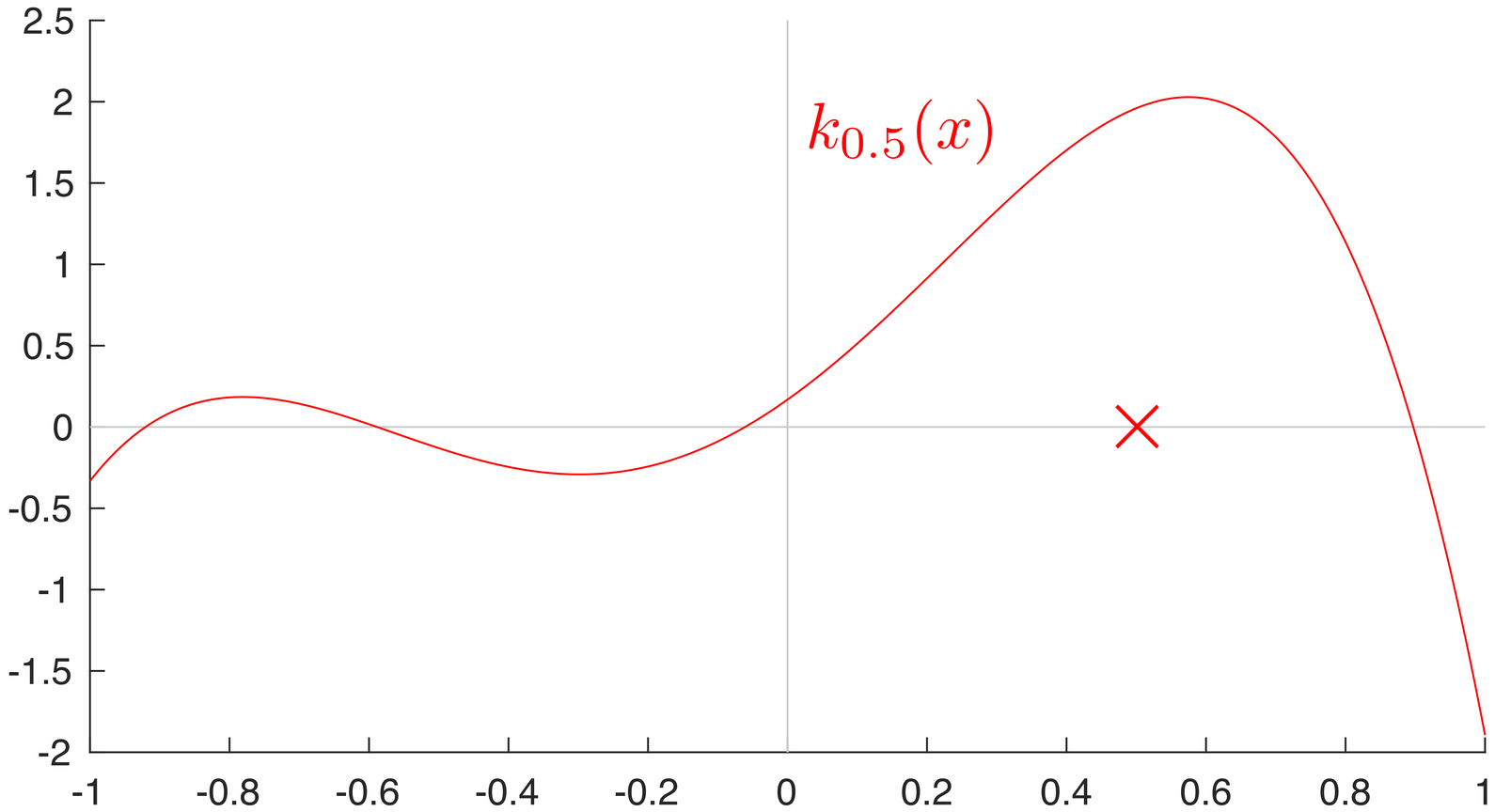
Example: $\mathcal{H}([-1,1]) = \text{span}_{n=1\cdots N}\{P_n(x)\}$



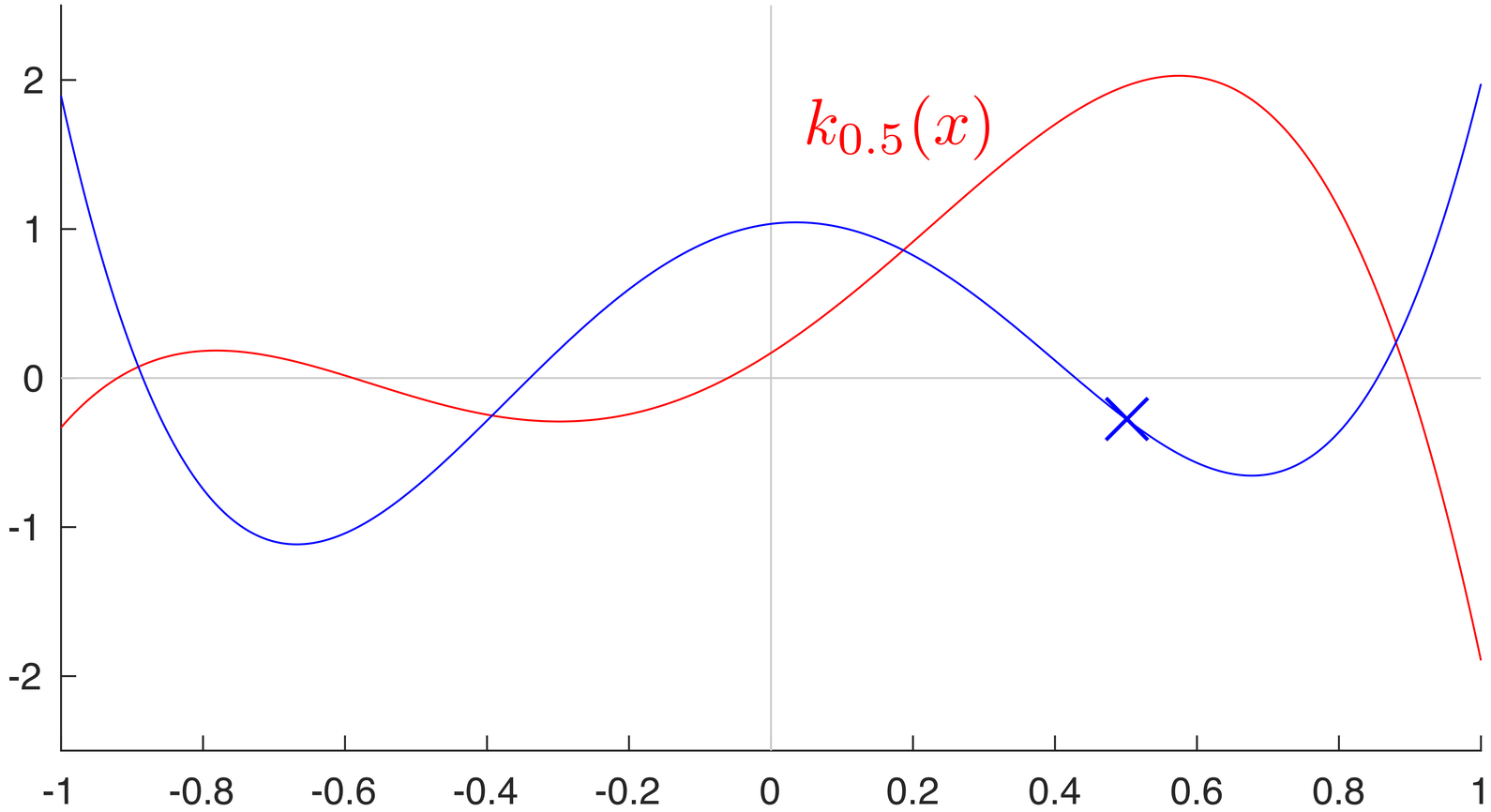
Example: $\mathcal{H}([-1,1]) = \mathrm{span}_{n=1\cdots N}\{P_n(x)\}$



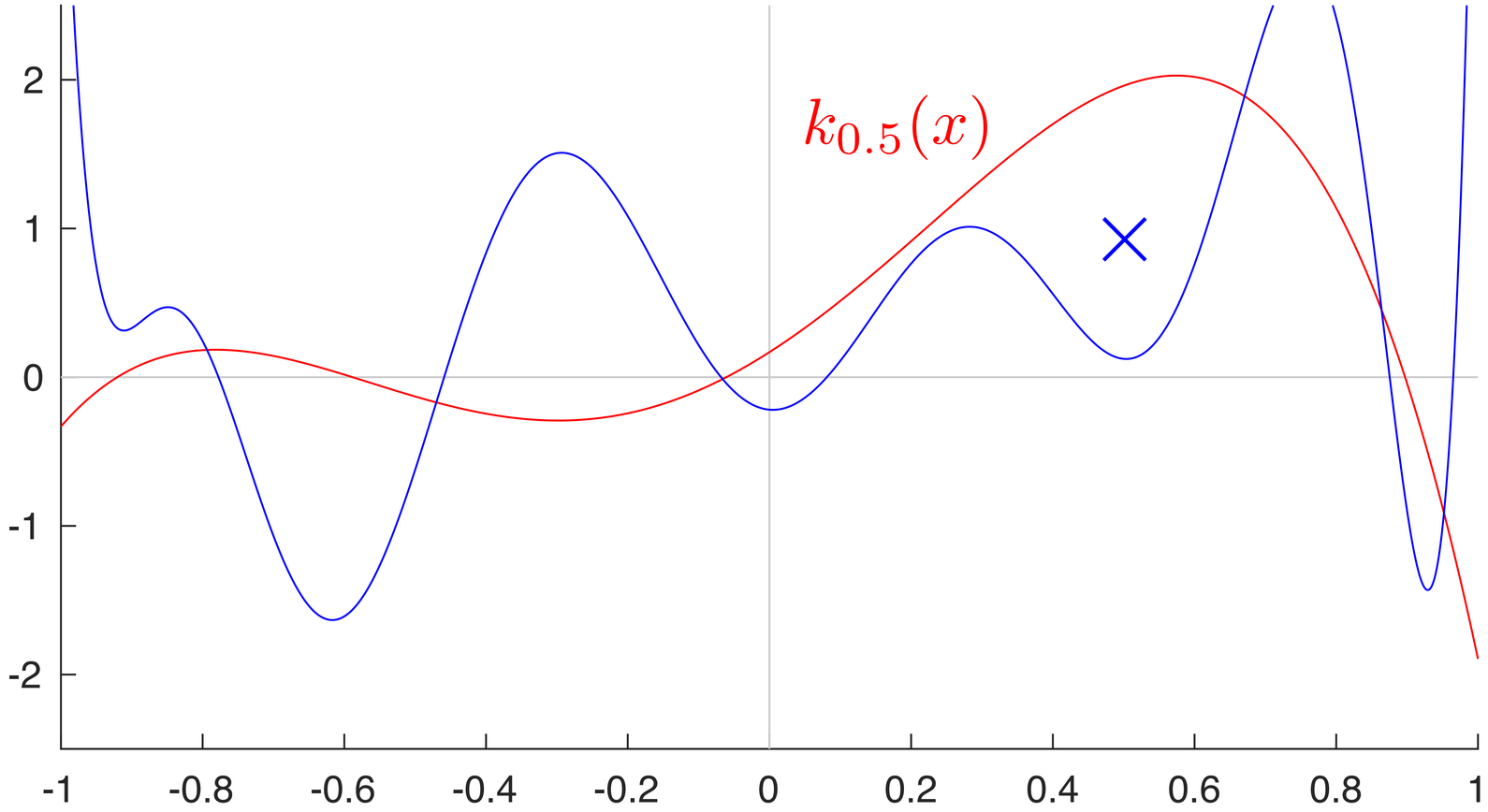
Example: $\mathcal{H}([-1,1]) = \text{span}_{n=1\dots N}\{P_n(x)\}$



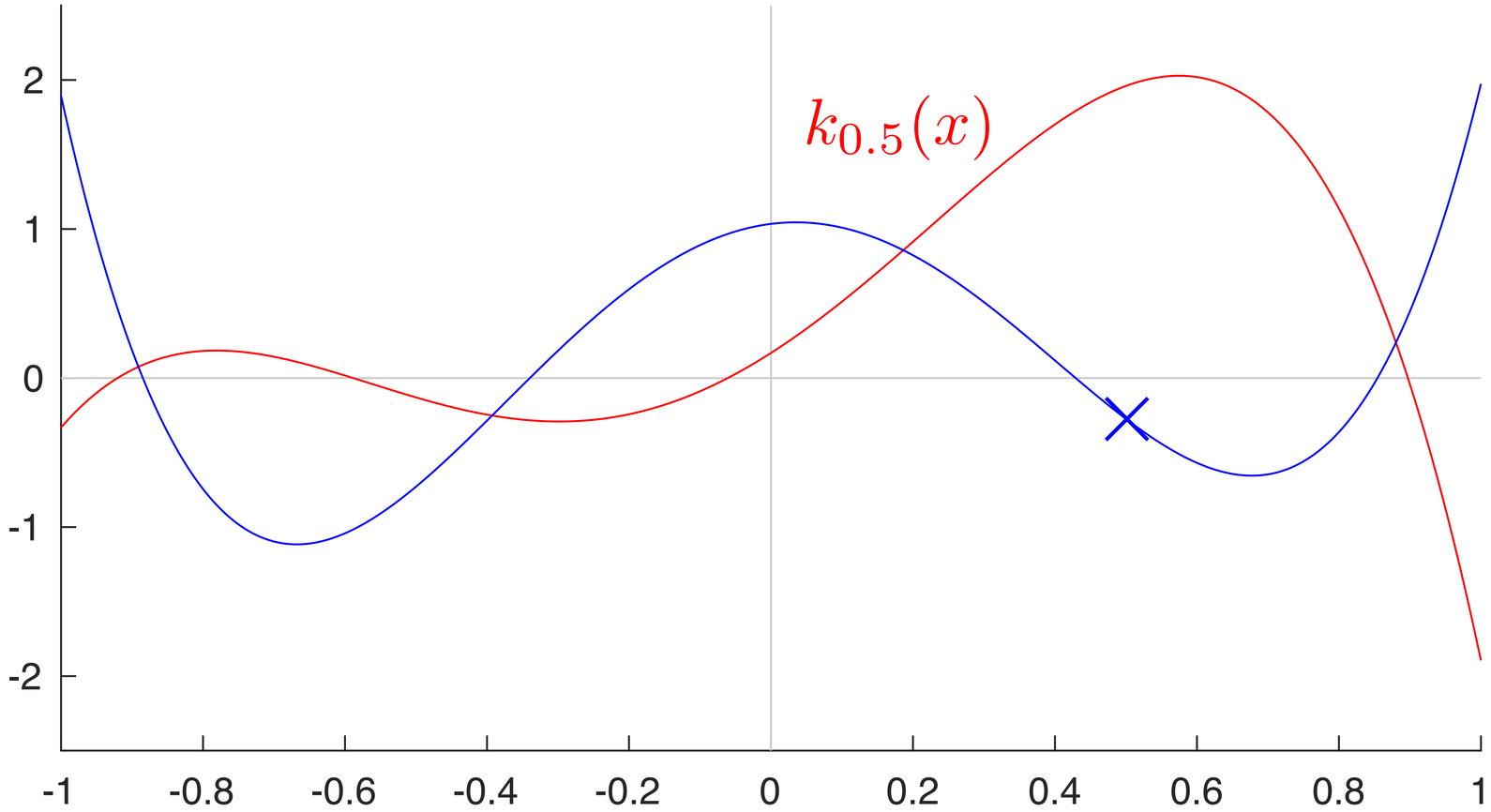
Example: $\mathcal{H}([-1,1]) = \text{span}_{n=1\dots N}\{P_n(x)\}$



Example: $\mathcal{H}([-1,1]) = \text{span}_{n=1\cdots N}\{P_n(x)\}$



Example: $\mathcal{H}([-1,1]) = \text{span}_{n=1\dots N}\{P_n(x)\}$



Reproducing kernel Hilbert spaces

$\mathcal{H}(X)$ is Hilbert and $\delta_{x_i}(f)$ continuous, then

$$\forall \bar{x} \in X, \exists k_{\bar{x}}(x) \in \mathcal{H}(X) \;\; : \;\; \left\langle k_{\bar{x}}(x) \, , \; f(x) \right\rangle = f(\bar{x})$$

Reproducing kernel Hilbert spaces

$\mathcal{H}(X)$ is Hilbert and $\delta_{x_i}(f)$ continuous, then

$$\forall \bar{x} \in X, \exists k_{\bar{x}}(x) \in \mathcal{H}(X) \; : \; \left\langle k_{\bar{x}}(x) \, , \, f(x) \right\rangle = f(\bar{x})$$

Reproducing kernel Hilbert spaces

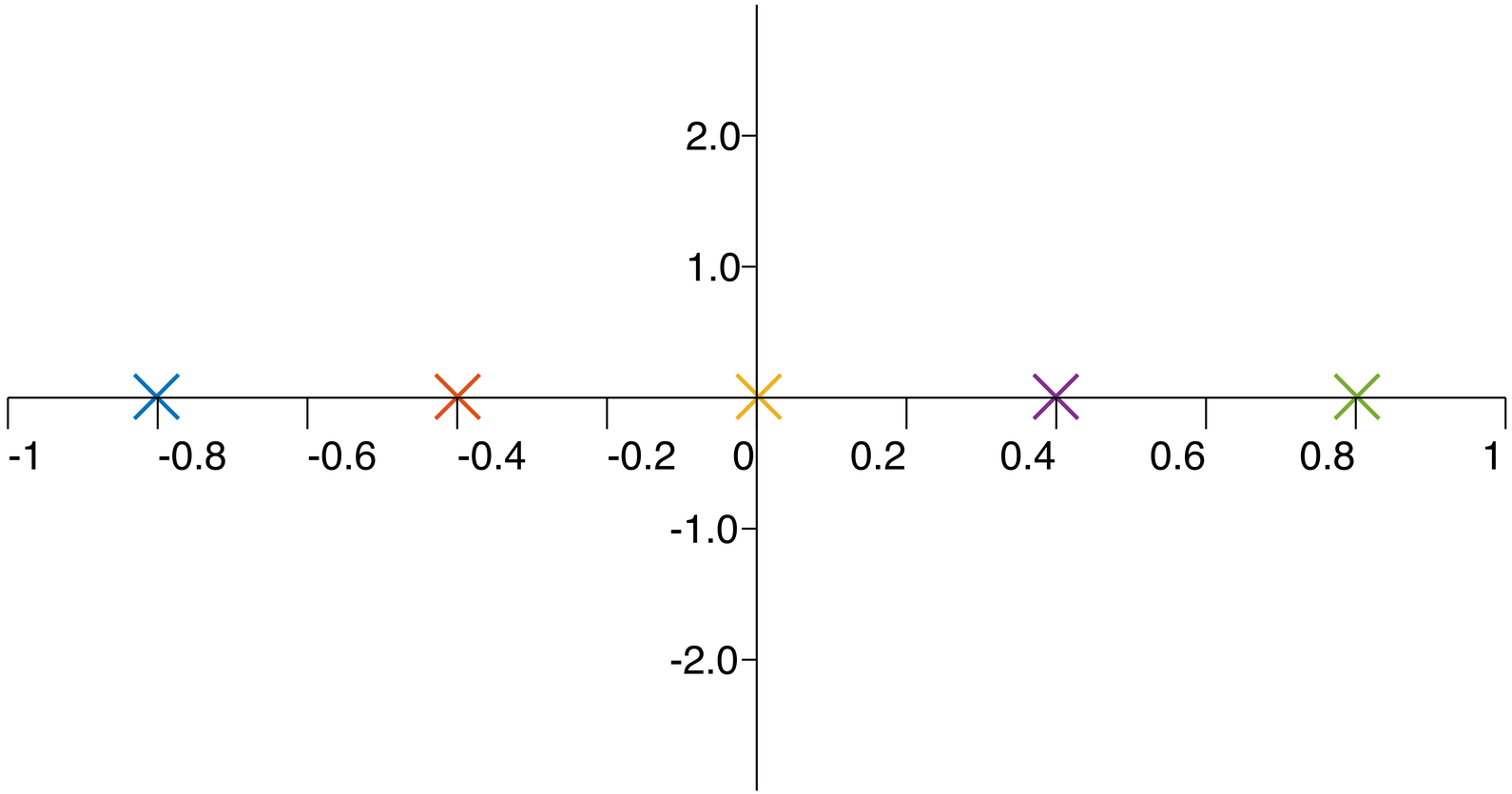
Let $\{\lambda_i\}$, $\lambda_i \in X$

Reproducing kernel Hilbert spaces

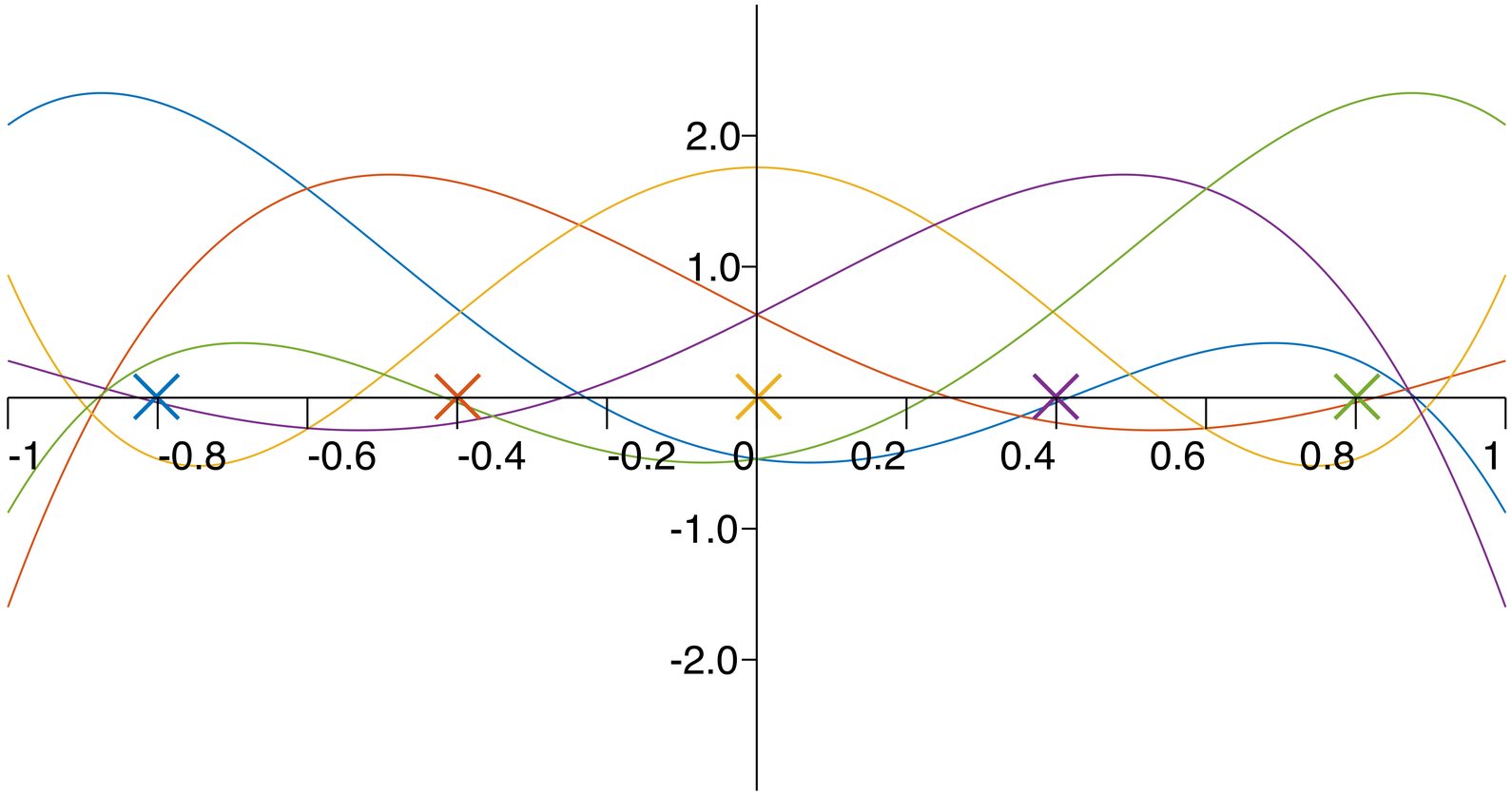
Let $\{\lambda_i\}$, $\lambda_i \in X$ s. t.

$$\text{span}\big(k_{\lambda_i}(x)\big) = \mathcal{H}(X)$$

Example: $\mathcal{H}([-1,1]) = \mathrm{span}_{n=1\cdots N}\{P_n(x)\}$



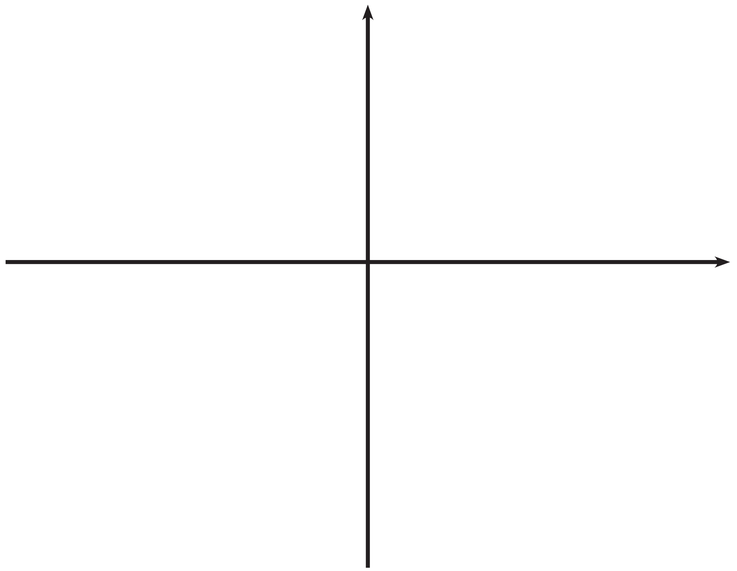
Example: $\mathcal{H}([-1,1]) = \mathrm{span}_{n=1\cdots N}\{P_n(x)\}$



Reproducing kernel Hilbert spaces

Let $\{\lambda_i\}$, $\lambda_i \in X$ s. t.

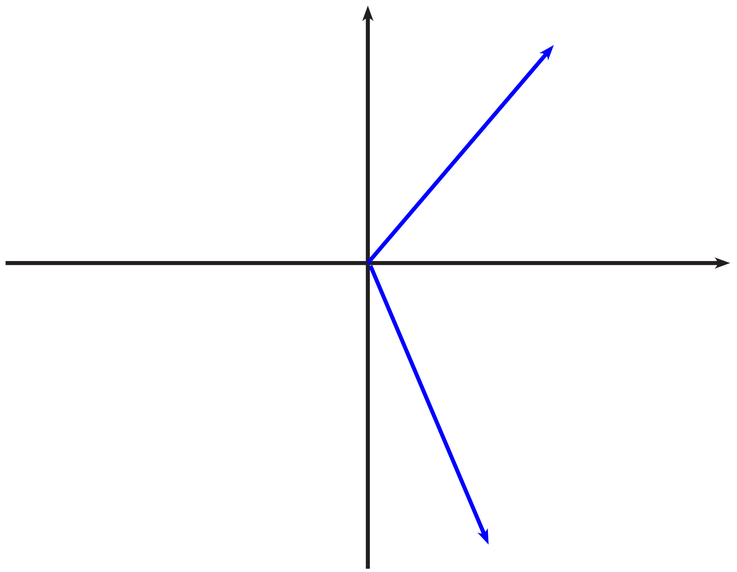
$$\text{span}\big(k_{\lambda_i}(x)\big) = \mathcal{H}(X)$$



Reproducing kernel Hilbert spaces

Let $\{\lambda_i\}$, $\lambda_i \in X$ s. t.

$$\text{span}\big(k_{\lambda_i}(x)\big) = \mathcal{H}(X)$$

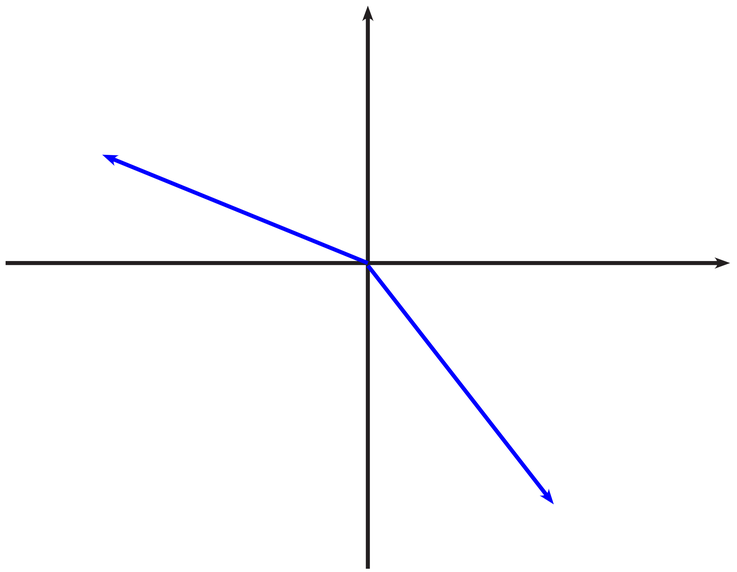


spanning set

Reproducing kernel Hilbert spaces

Let $\{\lambda_i\}$, $\lambda_i \in X$ s. t.

$$\text{span}\big(k_{\lambda_i}(x)\big) = \mathcal{H}(X)$$

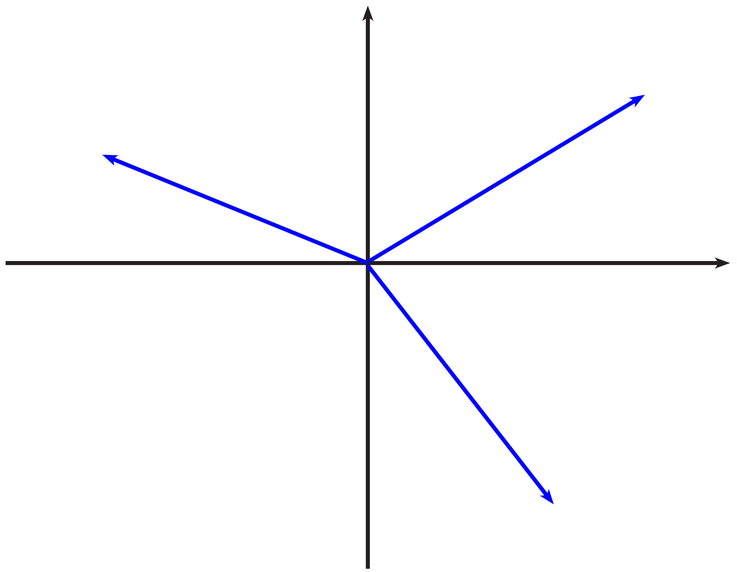


spanning set

Reproducing kernel Hilbert spaces

Let $\{\lambda_i\}$, $\lambda_i \in X$ s. t.

$$\mathrm{span}\big(k_{\lambda_i}(x)\big) = \mathcal{H}(X)$$

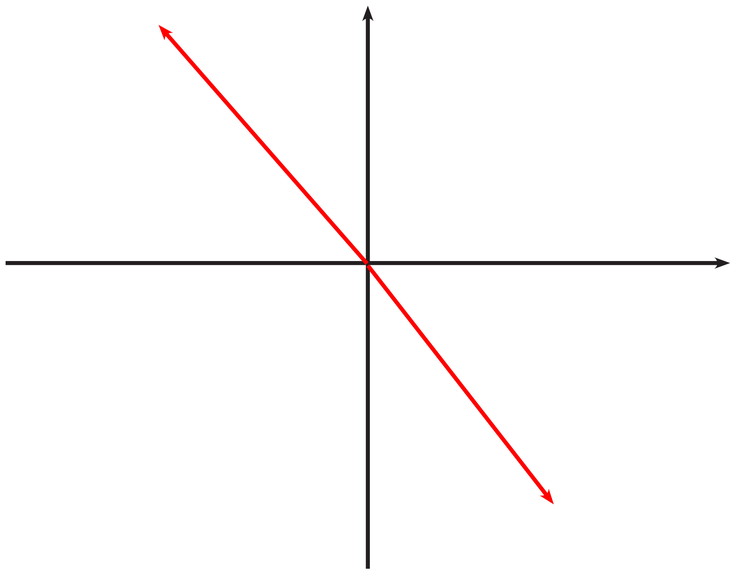


spanning set

Reproducing kernel Hilbert spaces

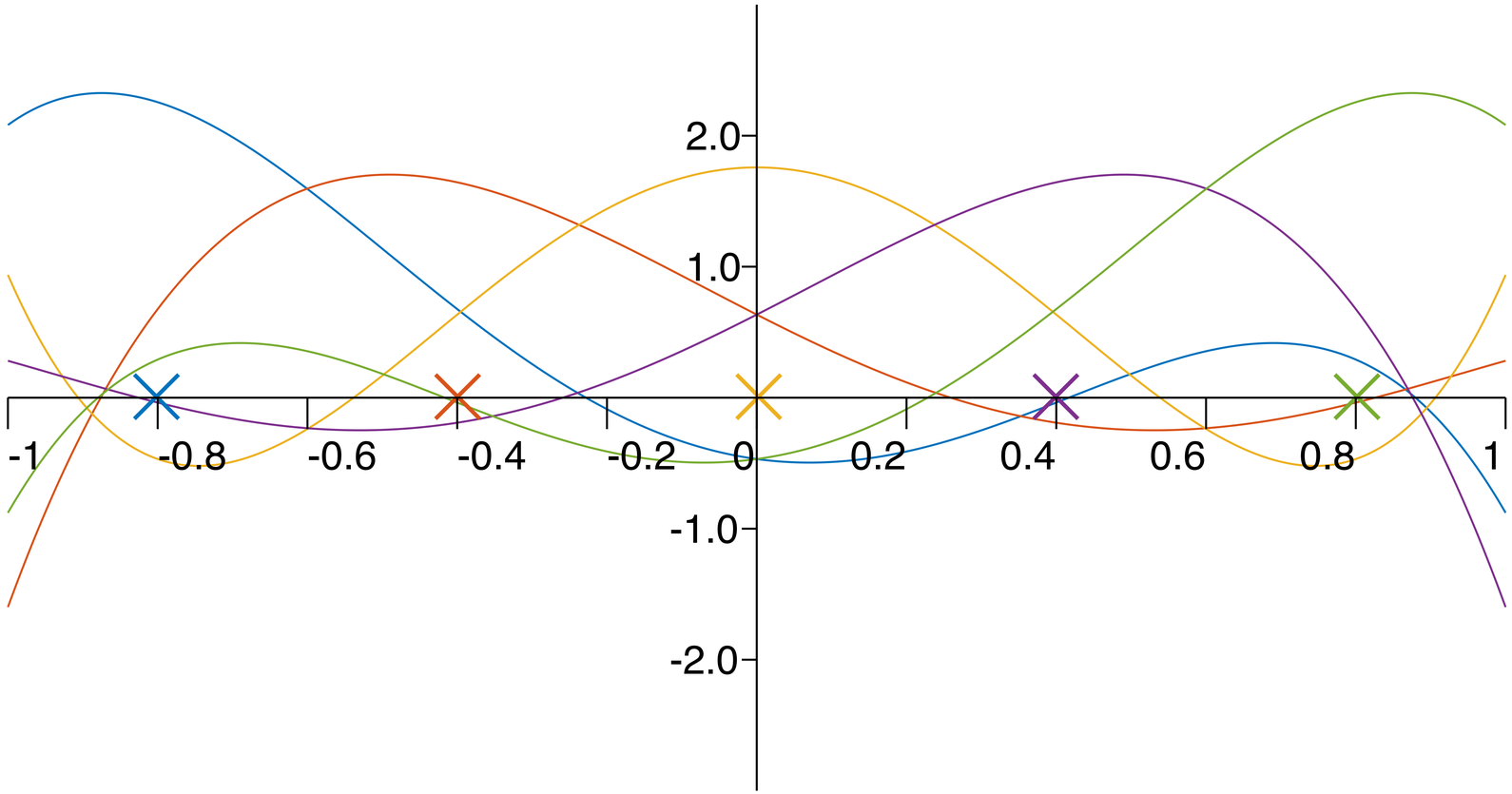
Let $\{\lambda_i\}$, $\lambda_i \in X$ s. t.

$$\text{span}\big(k_{\lambda_i}(x)\big) = \mathcal{H}(X)$$

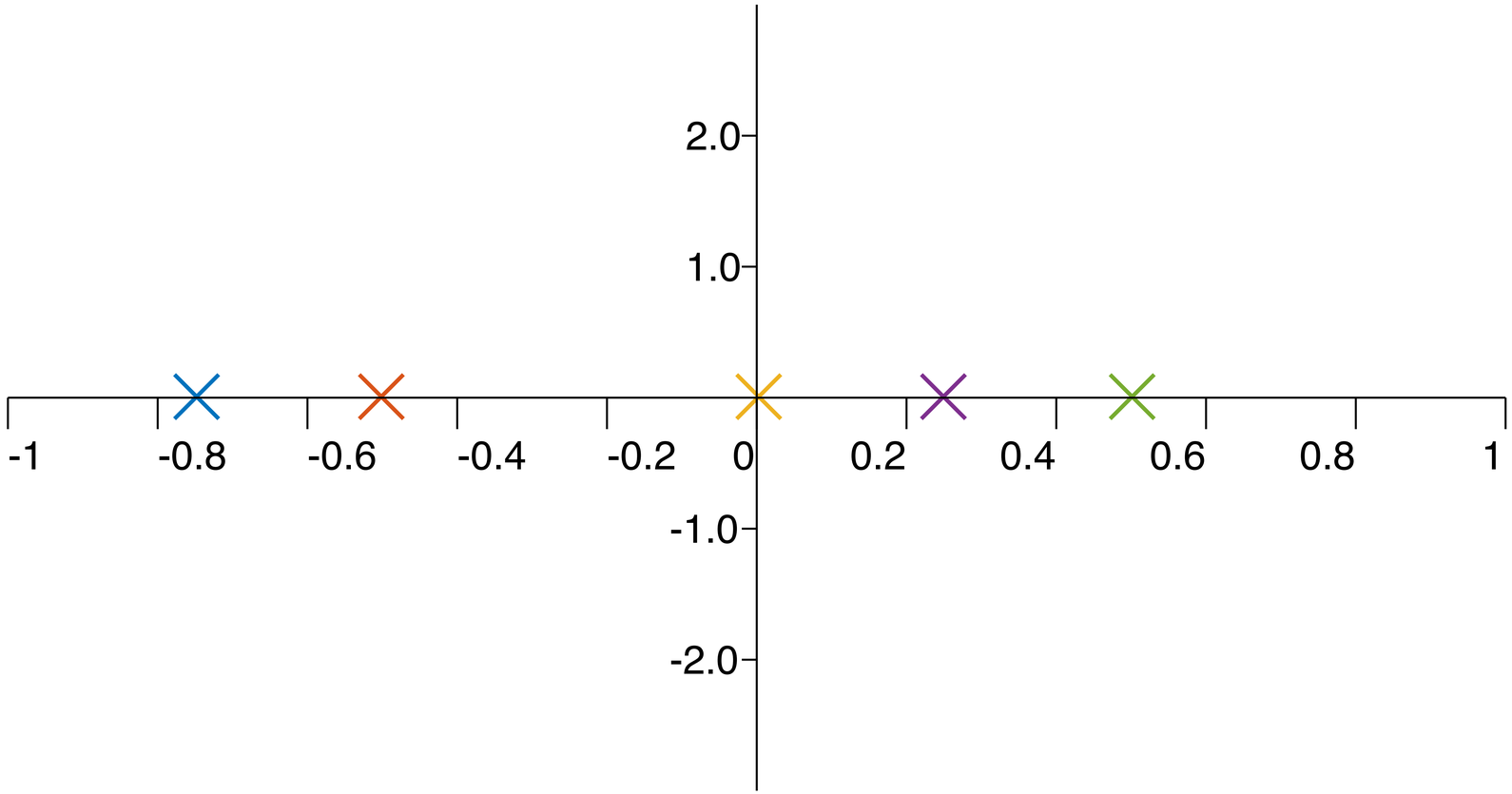


no spanning set

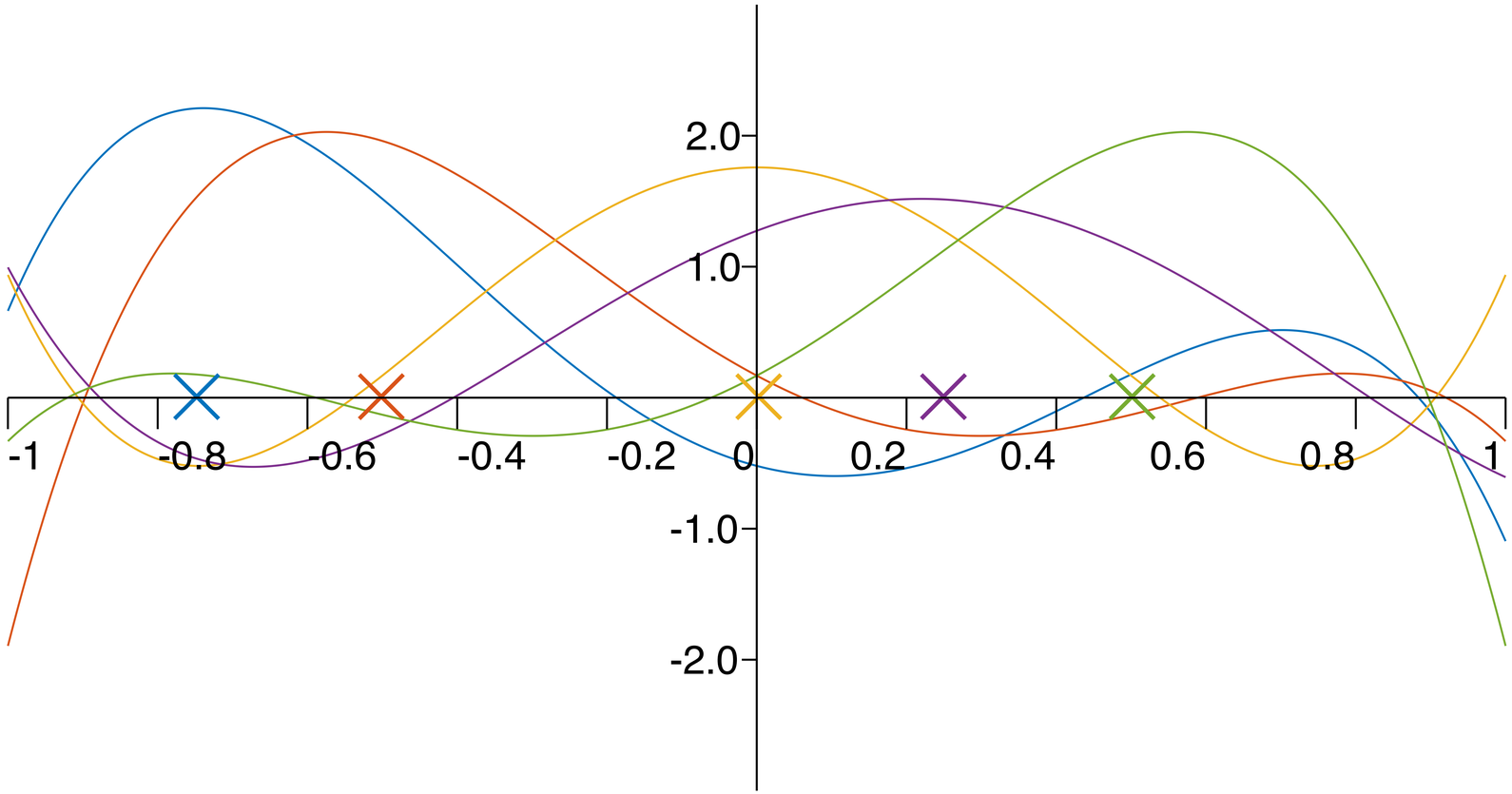
Example: $\mathcal{H}([-1,1]) = \mathrm{span}_{n=1\cdots N}\{P_n(x)\}$



Example: $\mathcal{H}([-1,1]) = \mathrm{span}_{n=1\cdots N}\{P_n(x)\}$



Example: $\mathcal{H}([-1,1]) = \mathrm{span}_{n=1\cdots N}\{P_n(x)\}$



Reproducing kernel Hilbert spaces

Let $\{\lambda_i\}$, $\lambda_i \in X$ s. t.

$$\text{span}\big(k_{\lambda_i}(x)\big) = \mathcal{H}(X)$$

Reproducing kernel Hilbert spaces

Let $\{\lambda_i\}$, $\lambda_i \in X$ s. t.

$$\text{span}\big(k_{\lambda_i}(x)\big) = \mathcal{H}(X)$$

$$\Updownarrow$$

$\{k_{\lambda_i}(x)\}$ forms a basis / frame for $\mathcal{H}(X)$

Reproducing kernel Hilbert spaces

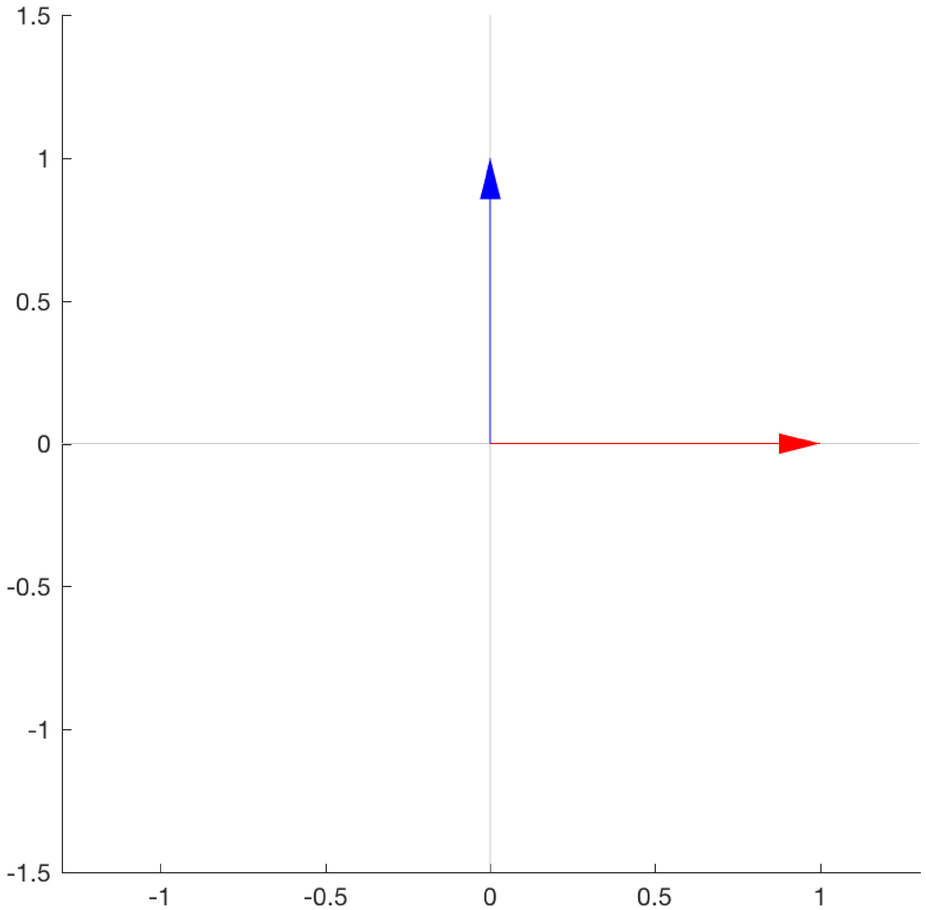
Let $\{\lambda_i\}$, $\lambda_i \in X$ s. t.

$$\text{span}\big(k_{\lambda_i}(x)\big) = \mathcal{H}(X)$$

$\{k_{\lambda_i}(x)\}$ forms a basis / frame for $\mathcal{H}(X)$

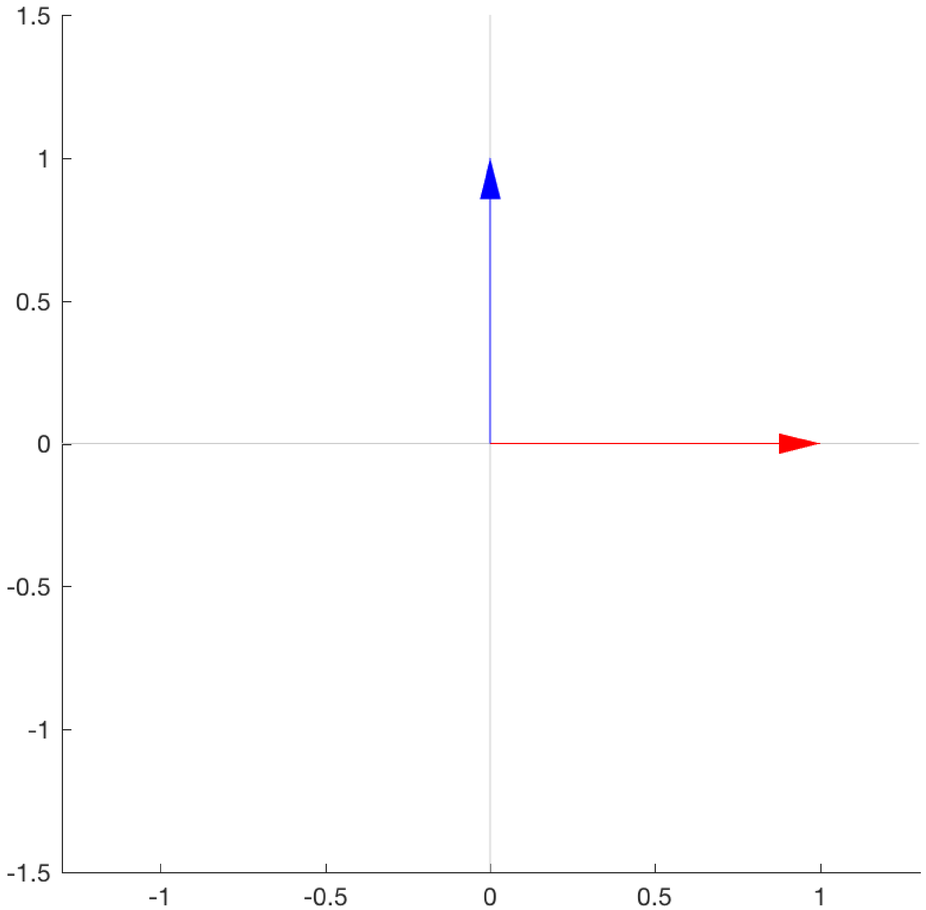
$$f(x) = \sum_{i=1}^{N} f_i \, k_{\lambda_i}(x)$$

Reproducing kernel Hilbert spaces



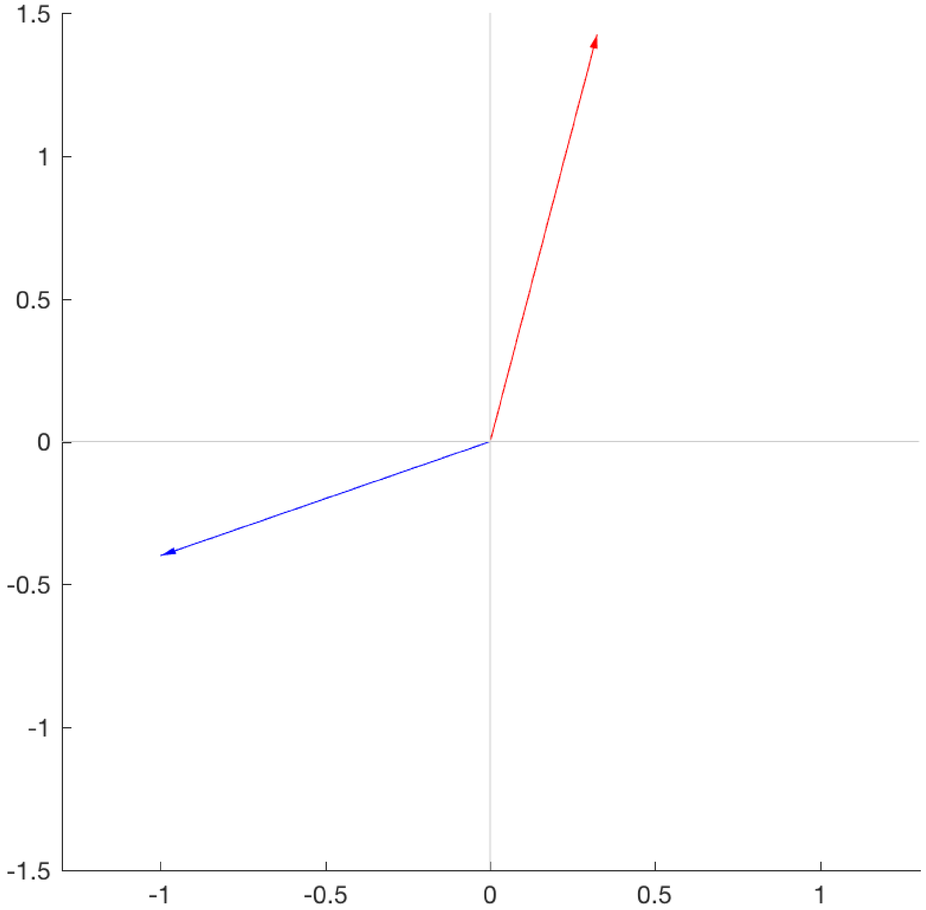
$$v = \sum_{i=1}^{2} v_i \, e_i$$

Reproducing kernel Hilbert spaces



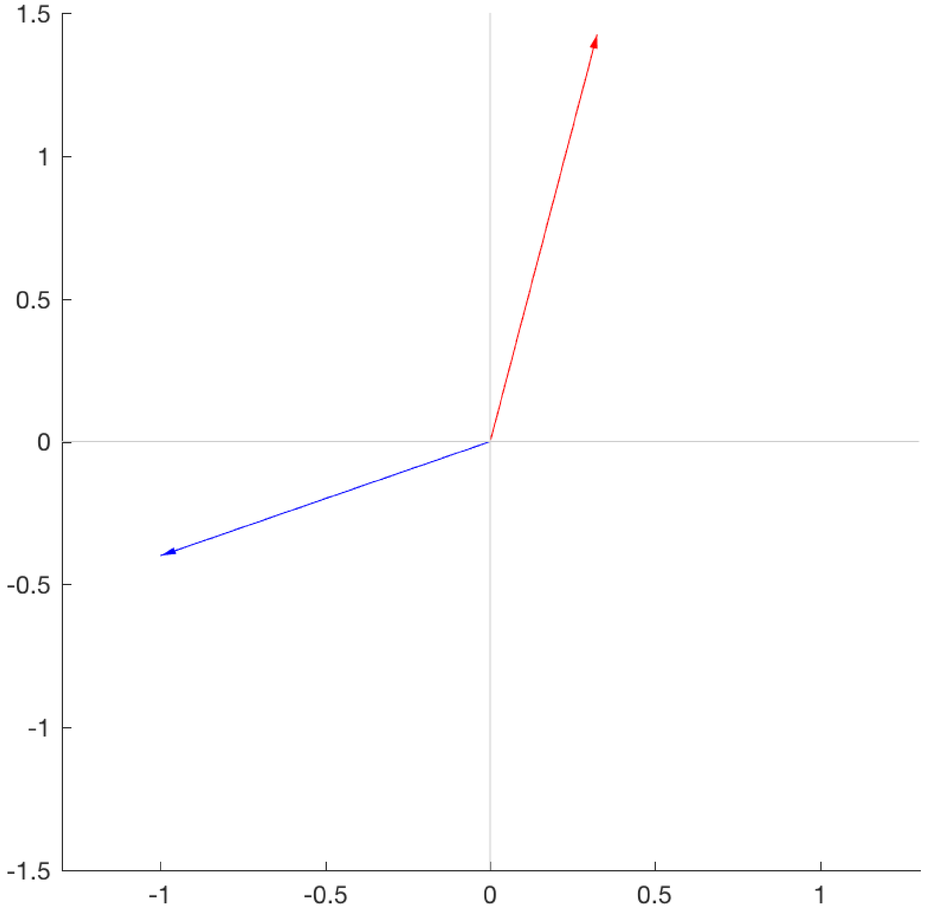
$$v = \sum_{i=1}^{2} \langle v \, , \, e_i \rangle \, e_i$$

Reproducing kernel Hilbert spaces



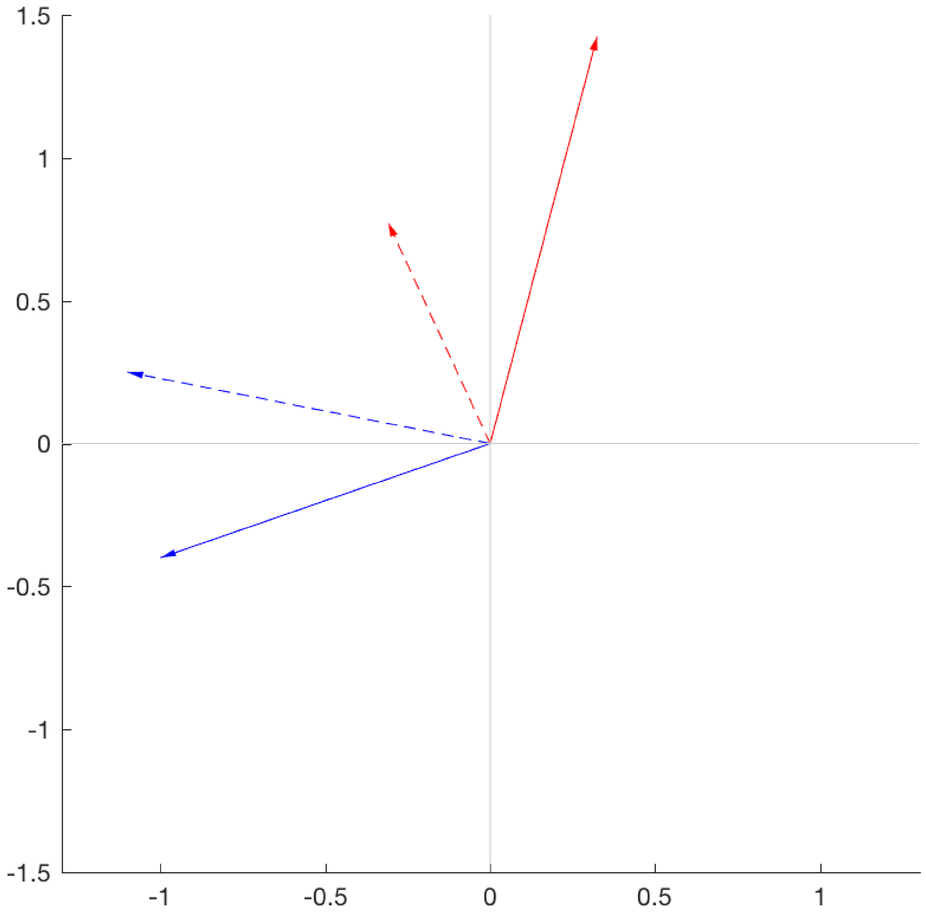
$$v = \sum_{i=1}^{2} v_i \, u_i$$

Reproducing kernel Hilbert spaces



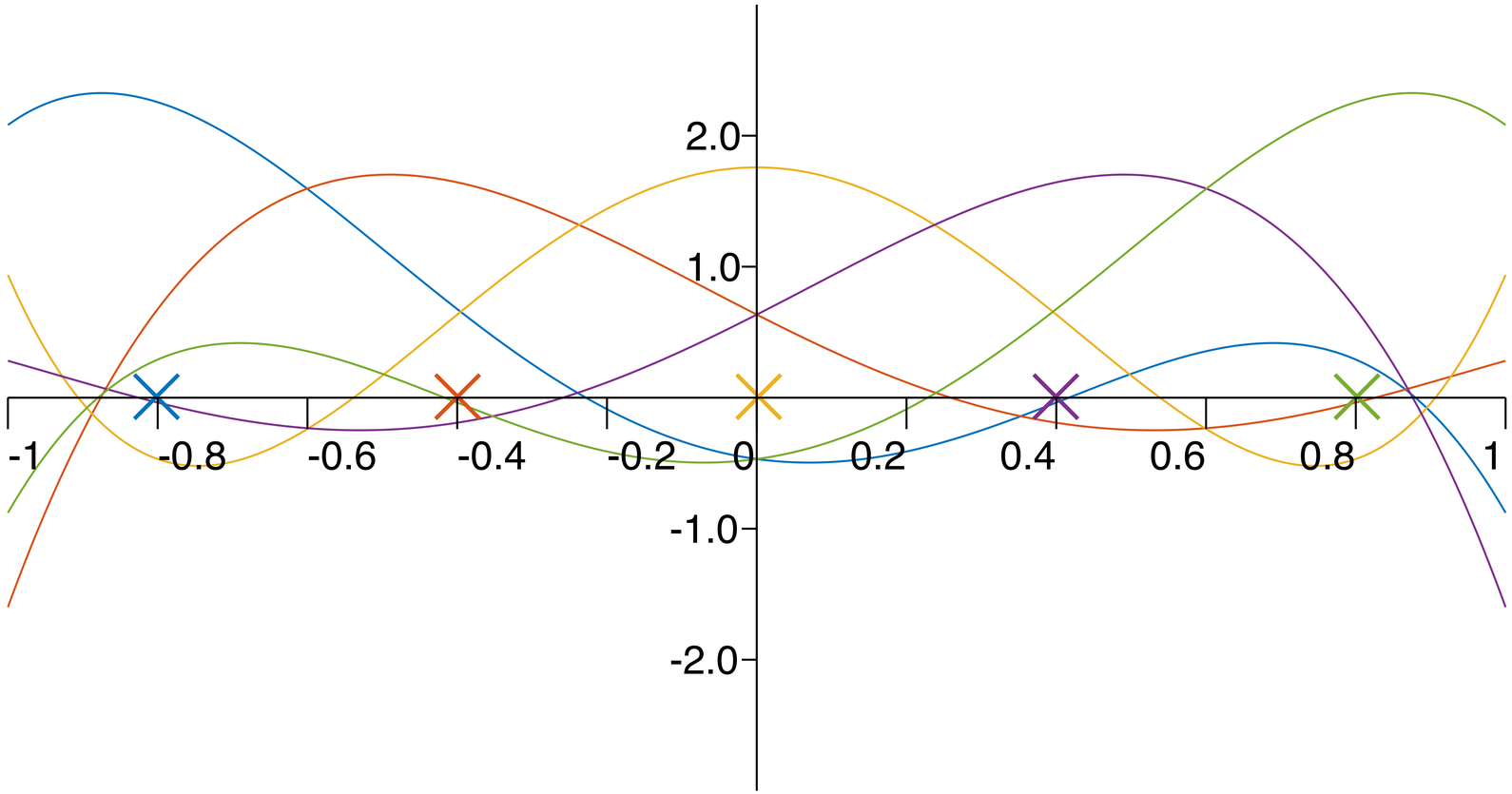
$$v = \sum_{i=1}^{2} \langle v \, , \, ? \, \rangle \, u_i$$

Reproducing kernel Hilbert spaces

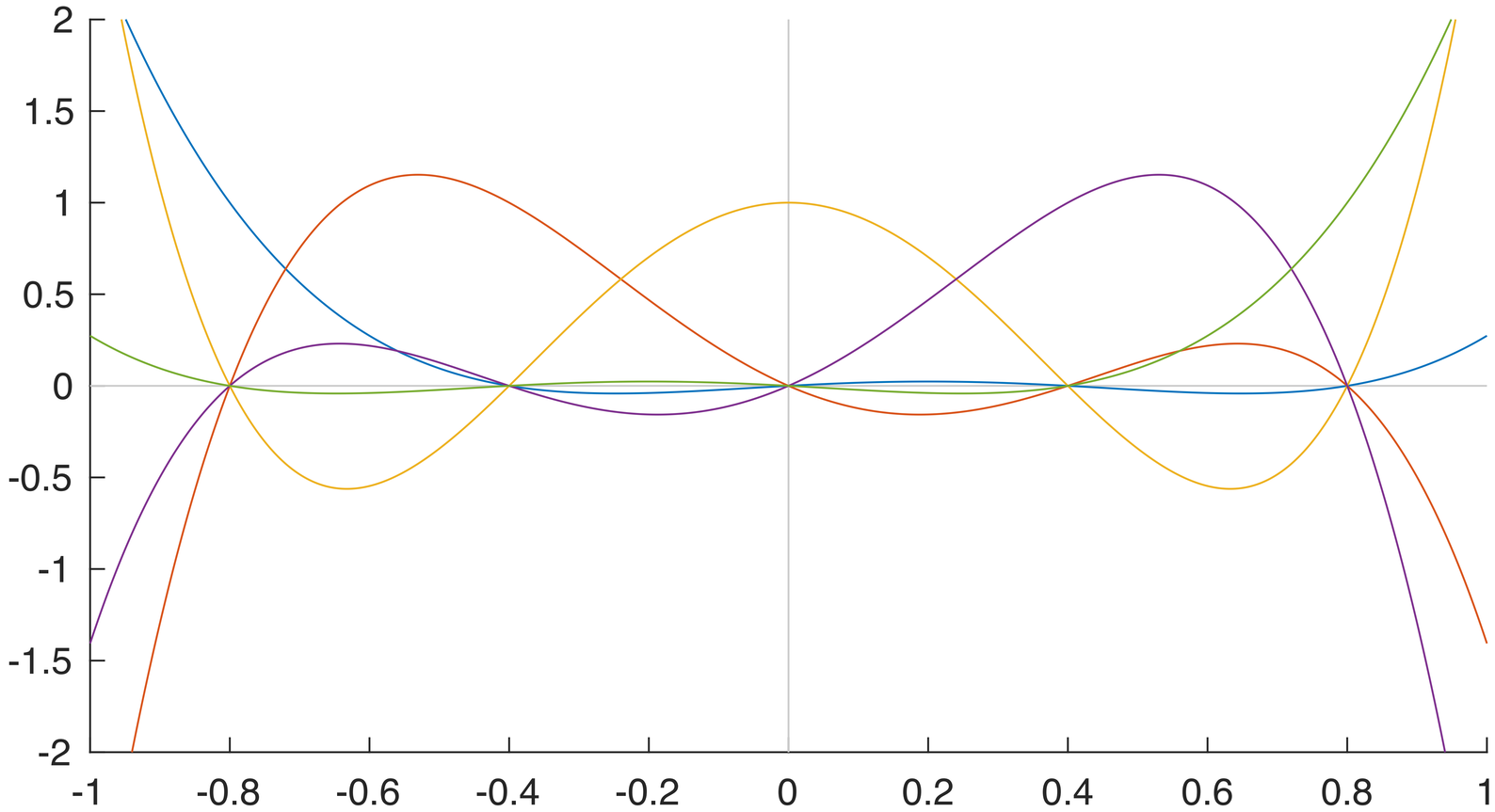


$$v = \sum_{i=1}^{2} \langle v \, , \, \tilde{u}_i \, \rangle \, u_i$$

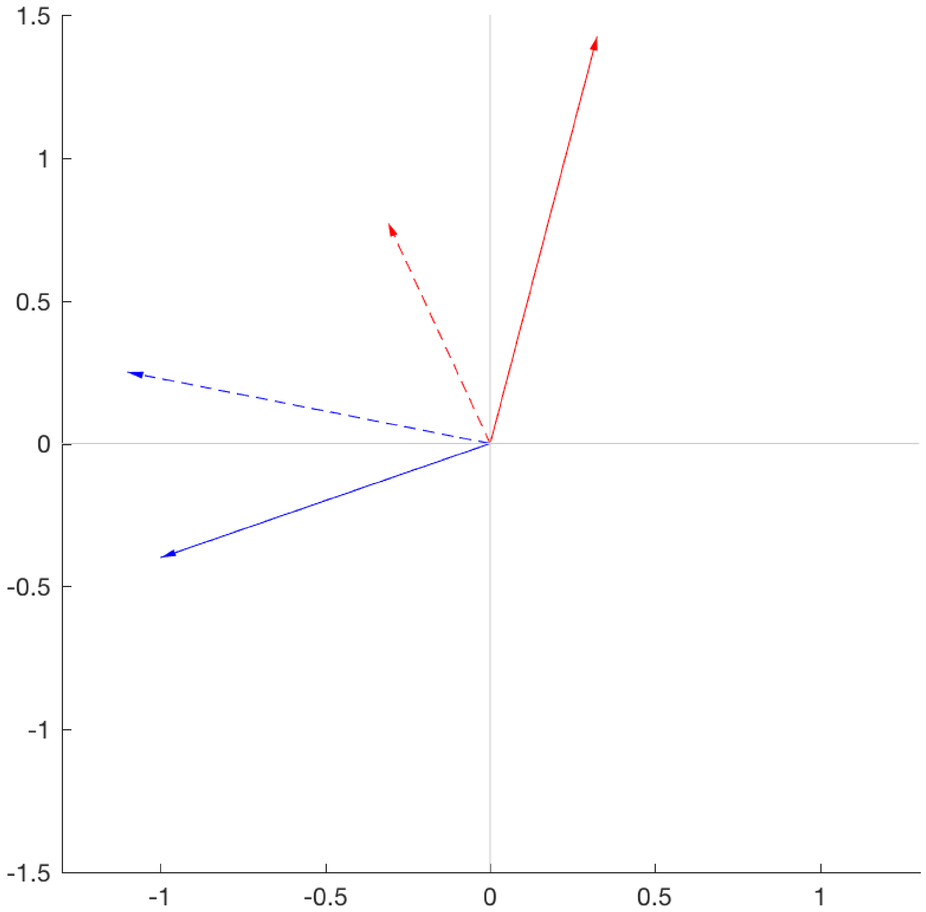
Example: $\mathcal{H}([-1,1]) = \mathrm{span}_{n=1\cdots N}\{P_n(x)\}$



Example: $\mathcal{H}([-1,1]) = \mathrm{span}_{n=1\cdots N}\{P_n(x)\}$

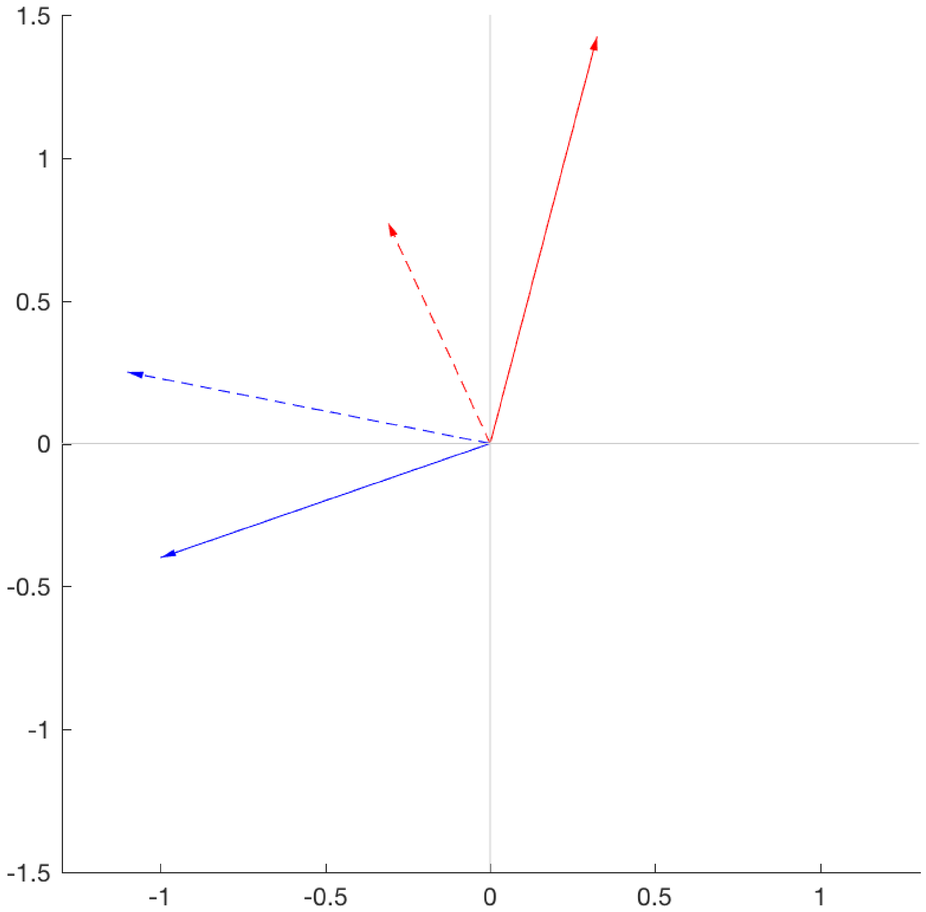


Reproducing kernel Hilbert spaces



$$v = \sum_{i=1}^{2} \langle v \, , \, \tilde{u}_i \, \rangle \, u_i$$

Reproducing kernel Hilbert spaces



$$v = \sum_{i=1}^{2} \langle v \, , \, u_i \, \rangle \, \tilde{u}_i$$

Reproducing kernel Hilbert spaces

Let $\{\lambda_i\}$, $\lambda_i \in X$ s. t. $\{k_{\lambda_i}(x)\}$ forms a basis / frame for $\mathcal{H}(X)$

$$f(x) = \sum_{i=1}^{N} f_i \, k_{\lambda_i}(x)$$

Reproducing kernel Hilbert spaces

Let $\{\lambda_i\}$, $\lambda_i \in X$ s. t. $\{k_{\lambda_i}(x)\}$ forms a basis / frame for $\mathcal{H}(X)$

$$f(x) = \sum_{i=1}^{N} f_i \, k_{\lambda_i}(x) = \sum_{i=1}^{N} \left\langle f(y), \tilde{k}_i(y) \right\rangle k_{\lambda_i}(x)$$

Reproducing kernel Hilbert spaces

Let $\{\lambda_i\}$, $\lambda_i \in X$ s. t. $\left\{k_{\lambda_i}(x)\right\}$ forms a basis / frame for $\mathcal{H}(X)$

$$f(x) = \sum_{i=1}^{N} f_i \, k_{\lambda_i}(x) = \sum_{i=1}^{N} \left\langle f(y), \tilde{k}_i(y) \right\rangle k_{\lambda_i}(x)$$

$$= \sum_{i=1}^{N} \left\langle f(y), k_{\lambda_i}(y) \right\rangle \tilde{k}_i(x)$$

Reproducing kernel Hilbert spaces

Let $\{\lambda_i\}$, $\lambda_i \in X$ s. t. $\{k_{\lambda_i}(x)\}$ forms a basis / frame for $\mathcal{H}(X)$

$$f(x) = \sum_{i=1}^{N} f_i \, k_{\lambda_i}(x) = \sum_{i=1}^{N} \left\langle f(y), \tilde{k}_i(y) \right\rangle k_{\lambda_i}(x)$$

$$= \sum_{i=1}^{N} \boxed{\left\langle f(y), k_{\lambda_i}(y) \right\rangle} \tilde{k}_i(x)$$

Reproducing kernel Hilbert spaces

Let $\{\lambda_i\}$, $\lambda_i \in X$ s. t. $\{k_{\lambda_i}(x)\}$ forms a basis / frame for $\mathcal{H}(X)$

$$f(x) = \sum_{i=1}^{N} f_i \, k_{\lambda_i}(x) = \sum_{i=1}^{N} \left\langle f(y), \tilde{k}_i(y) \right\rangle k_{\lambda_i}(x)$$

$$= \sum_{i=1}^{N} \boxed{\left\langle f(y), k_{\lambda_i}(y) \right\rangle} \tilde{k}_i(x)$$

$$= f(\lambda_i)$$

Reproducing kernel Hilbert spaces

Let $\{\lambda_i\}$, $\lambda_i \in X$ s. t. $\{k_{\lambda_i}(x)\}$ forms a basis / frame for $\mathcal{H}(X)$

$$f(x) = \sum_{i=1}^{N} f_i \, k_{\lambda_i}(x) = \sum_{i=1}^{N} \left\langle f(y), \tilde{k}_i(y) \right\rangle k_{\lambda_i}(x)$$

$$= \sum_{i=1}^{N} \underbrace{\left\langle f(y), k_{\lambda_i}(y) \right\rangle}_{= f(\lambda_i)} \tilde{k}_i(x) = \sum_{i=1}^{N} f(\lambda_i) \, \tilde{k}_i(x)$$

Reproducing kernel Hilbert spaces

Let $\{\lambda_i\}$, $\lambda_i \in X$ s. t. $\{k_{\lambda_i}(x)\}$ forms a basis / frame for $\mathcal{H}(X)$

$$f(x) = \sum_{i=1}^{N} f(\lambda_i)\, \tilde{k}_i(x)$$

Reproducing kernel Hilbert spaces

Let $\{\lambda_i\}$, $\lambda_i \in X$ s. t. $\left\{k_{\lambda_i}(x)\right\}$ forms a basis / frame for $\mathcal{H}(X)$

$$f(x) = \sum_{i=1}^{N} f(\lambda_i)\, \tilde{k}_i(x)$$

reproducing kernel frame

Reproducing kernel Hilbert spaces

Let $\{\lambda_i\}$, $\lambda_i \in X$ s. t. $\{k_{\lambda_i}(x)\}$ forms a basis / frame for $\mathcal{H}(X)$

$$f(x) = \sum_{i=1}^{N} f(\lambda_i)\, \tilde{k}_i(x)$$

reproducing kernel frame $\Leftrightarrow$ sampling theorem

Reproducing kernel Hilbert spaces

Let $\{\lambda_i\}$, $\lambda_i \in X$ s. t. $\{k_{\lambda_i}(x)\}$ forms a basis / frame for $\mathcal{H}(X)$

$$f(x) = \sum_{i=1}^{N} f(\lambda_i)\, \tilde{k}_i(x)$$

reproducing kernel frame $\Leftrightarrow$ sampling theorem

Shannon: $$f(x) = \sum_{i=-\infty}^{\infty} f(i)\, \mathrm{sinc}(x - i)$$

Reproducing kernel Hilbert spaces

Let $\{\lambda_i\}$, $\lambda_i \in X$ s. t. $\{k_{\lambda_i}(x)\}$ forms a basis / frame for $\mathcal{H}(X)$

$$f(x) = \sum_{i=1}^{N} f(\lambda_i)\, \tilde{k}_i(x)$$

reproducing kernel frame $\Leftrightarrow$ sampling theorem

Shannon: $$f(x) = \sum_{i=-\infty}^{\infty} f(i)\, \mathrm{sinc}(x-i)$$

$$= \sum_{i=-\infty}^{\infty} \big\langle f(y), \mathrm{sinc}(y-i) \big\rangle\, \mathrm{sinc}(x-i)$$

Reproducing kernel Hilbert spaces

Let $\{\lambda_i\}$, $\lambda_i \in X$ s. t. $\{k_{\lambda_i}(x)\}$ forms a basis / frame for $\mathcal{H}(X)$

$$f(x) = \sum_{i=1}^{N} f(\lambda_i)\, \tilde{k}_i(x)$$

reproducing kernel frame $\Leftrightarrow$ sampling theorem

Reproducing kernel Hilbert spaces

Let $\{\lambda_i\}$, $\lambda_i \in X$ s. t. $\{k_{\lambda_i}(x)\}$ forms a basis / frame for $\mathcal{H}(X)$

$$f(x) = \sum_{i=1}^{N} f(\lambda_i)\, \tilde{k}_i(x)$$

reproducing kernel frame $\Leftrightarrow$ sampling theorem

- arbitrary locations

Reproducing kernel Hilbert spaces

Let $\{\lambda_i\}$, $\lambda_i \in X$ s. t. $\left\{k_{\lambda_i}(x)\right\}$ forms a basis / frame for $\mathcal{H}(X)$

$$f(x) = \sum_{i=1}^{N} f(\lambda_i)\, \tilde{k}_i(x)$$

reproducing kernel frame $\Leftrightarrow$ sampling theorem

- arbitrary locations
- non-bandlimited functions

Reproducing kernel Hilbert spaces

Let $\{\lambda_i\}$, $\lambda_i \in X$ s. t. $\left\{k_{\lambda_i}(x)\right\}$ forms a basis / frame for $\mathcal{H}(X)$

$$f(x) = \sum_{i=1}^{N} f(\lambda_i)\, \tilde{k}_i(x)$$

reproducing kernel frame $\Leftrightarrow$ sampling theorem

- arbitrary locations
- non-bandlimited functions
- over-sampling

Reproducing kernel Hilbert spaces

Let $\{\lambda_i\}$, $\lambda_i \in X$ s. t. $\left\{k_{\lambda_i}(x)\right\}$ forms a basis / frame for $\mathcal{H}(X)$

$$f(x) = \sum_{i=1}^{N} f(\lambda_i)\, \tilde{k}_i(x)$$

reproducing kernel frame $\Leftrightarrow$ sampling theorem

- arbitrary locations
- non-bandlimited functions
- over-sampling
- arbitrary domains

Reproducing kernel Hilbert spaces

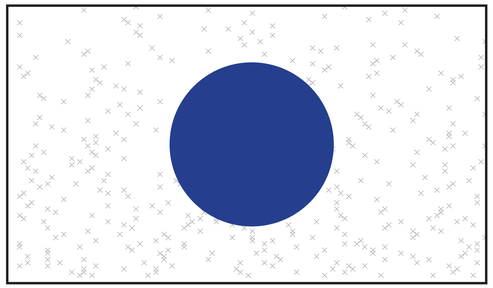
Let $\{\lambda_i\}$, $\lambda_i \in X$ s. t. $\left\{k_{\lambda_i}(x)\right\}$ forms a basis / frame for $\mathcal{H}(X)$

$$f(x) = \sum_{i=1}^{N} f(\lambda_i)\, \tilde{k}_i(x)$$

reproducing kernel frame $\Leftrightarrow$ sampling theorem

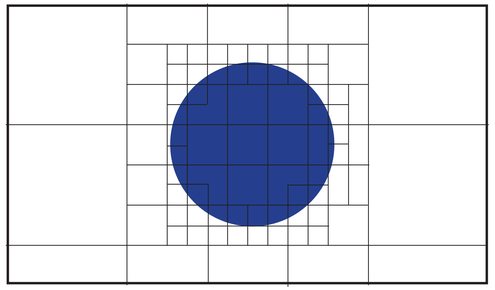
- arbitrary locations
- non-bandlimited functions
- over-sampling
- arbitrary domains
- optimal locations for setting

But what has this to do with sampling?



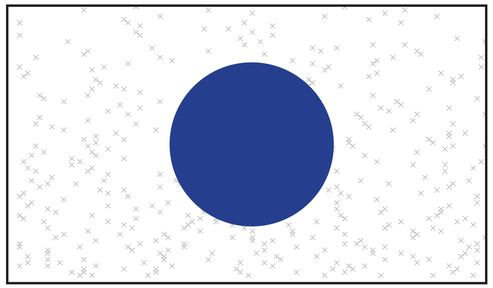
$$\bar{f}_i = B^+ \bar{f}(x_i)$$

$$\bar{f}(x_i) = B\, \bar{f}_i$$



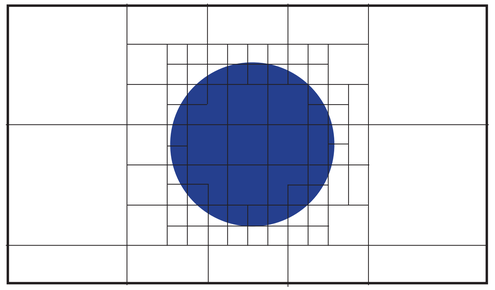
$$\left\{ f(x_i) \right\} \in \mathbb{R}^m \qquad \left\{ f_i \right\} \in \mathbb{R}^n$$

But what has this to do with sampling?



$$\bar{f}_i = B^+ \bar{f}(x_i)$$

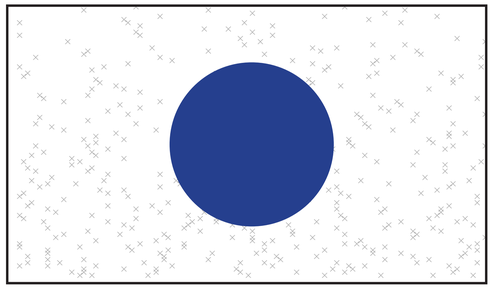
$$\bar{f}(x_i) = B\, \bar{f}_i$$



$$\left\{ f(x_i) \right\} \in \mathbb{R}^m \qquad \left\{ f_i \right\} \in \mathbb{R}^n$$

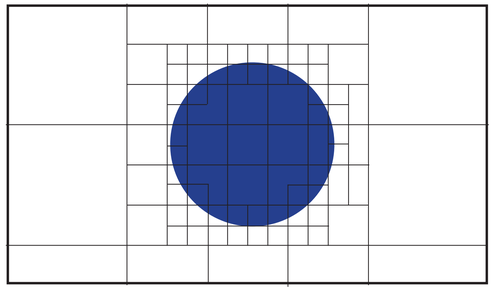
$$f(x) = \sum_{i=1}^{N} f(\lambda_i)\, \tilde{k}_i(x)$$

But what has this to do with sampling?



$$\bar{f}_i = B^+ \bar{f}(x_i)$$

$$\bar{f}(x_i) = B\, \bar{f}_i$$



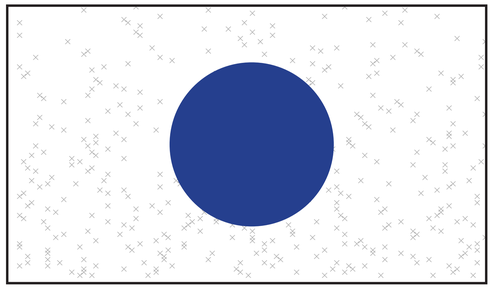
$$\left\{ f(x_i) \right\} \in \mathbb{R}^m$$

$$\left\{ f_i \right\} \in \mathbb{R}^n$$

$$f(x) = \sum_{i=1}^{N} f(\lambda_i)\, \tilde{k}_i(x)$$

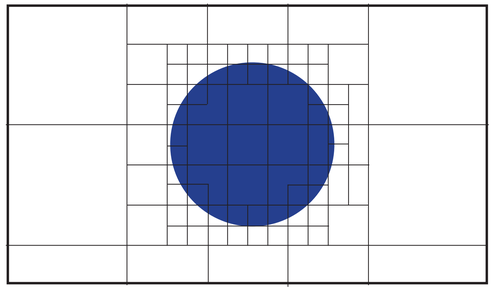
$$f(x) = \sum_{i=1}^{N} f_i\, \phi_i(x)$$

But what has this to do with sampling?



$$\bar{f}_i = B^+ \bar{f}(x_i)$$

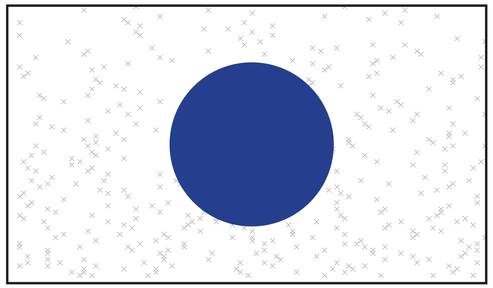
$$\bar{f}(x_i) = B\, \bar{f}_i$$



$$\Big\{ f(x_i) \Big\} \in \mathbb{R}^m \qquad \Big\{ f_i \Big\} \in \mathbb{R}^n$$

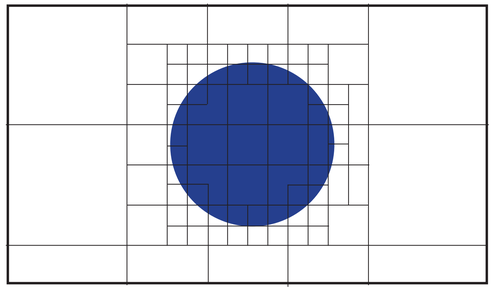
$$f(x) = \sum_{i=1}^{N} f(\lambda_i)\, \tilde{k}_i(x) \quad \xrightarrow{\text{change of basis}} \quad f(x) = \sum_{i=1}^{N} f_i\, \phi_i(x)$$

But what has this to do with sampling?



$$\bar{f}_i = B^+ \bar{f}(x_i)$$

$$\bar{f}(x_i) = B\, \bar{f}_i$$



$$\left\{ f(x_i) \right\} \in \mathbb{R}^m$$

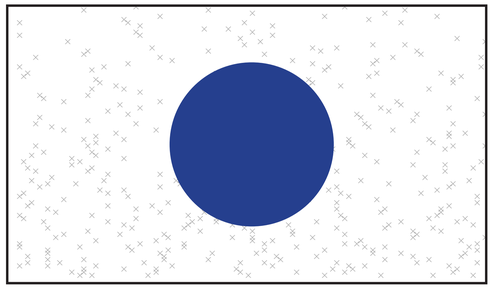
$$\left\{ f_i \right\} \in \mathbb{R}^n$$

$$f(x) = \sum_{i=1}^{N} f(\lambda_i)\, \tilde{k}_i(x)$$

change of basis

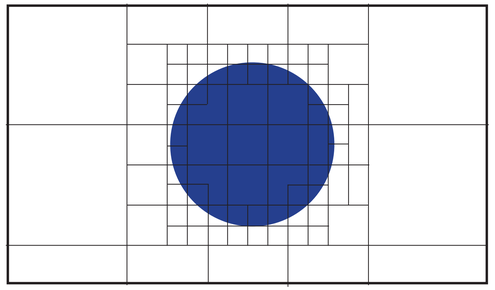
$$f(x) = \sum_{i=1}^{N} f_i\, \phi_i(x)$$

But what has this to do with sampling?



$$\bar{f}_i = B^+ \bar{f}(x_i)$$

$$\bar{f}(x_i) = B\, \bar{f}_i$$



$$\left\{ f(x_i) \right\} \in \mathbb{R}^m$$

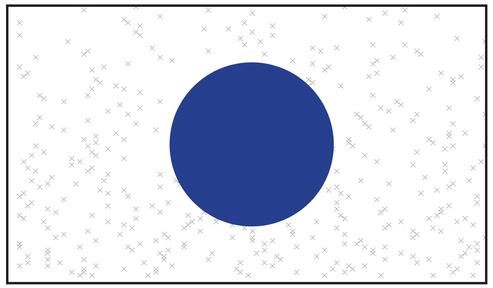
$$\left\{ f_i \right\} \in \mathbb{R}^n$$

$$f(x) = \sum_{i=1}^{N} f(\lambda_i)\, \tilde{k}_i(x)$$

change of basis

$$f(x) = \sum_{i=1}^{N} f_i\, \phi_i(x)$$

But what has this to do with sampling?

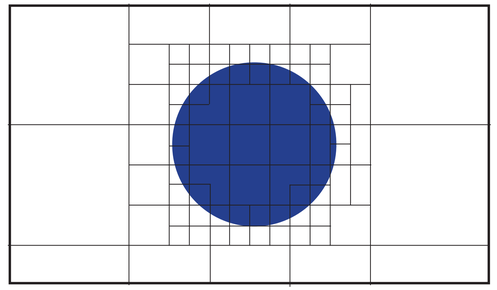


$$\left\{ f(x_i) \right\} \in \mathbb{R}^m$$

$$\bar{f}_i = B^+ \bar{f}(x_i)$$

$$\bar{f}(x_i) = B\, \bar{f}_i$$

sampling theorem for sparse image representation



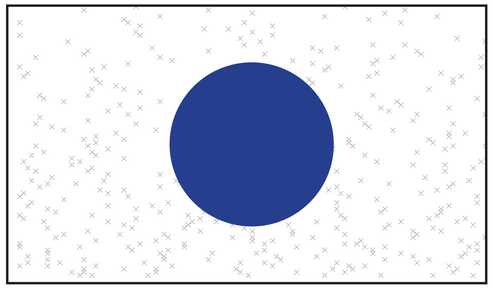
$$\left\{ f_i \right\} \in \mathbb{R}^n$$

$$f(x) = \sum_{i=1}^{N} f(\lambda_i)\, \tilde{k}_i(x)$$

change of basis

$$f(x) = \sum_{i=1}^{N} f_i\, \phi_i(x)$$

But what has this to do with sampling?

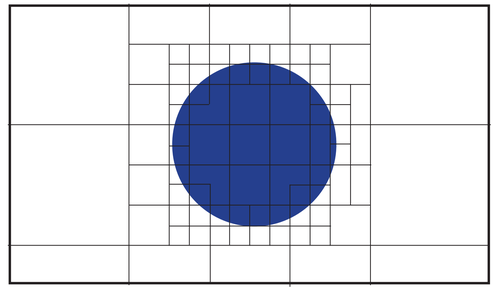


$$f(x) = \sum_{i=1}^{N} f(\lambda_i)\, \tilde{k}_i(x)$$

$$\bar{f}_i = B^+ \bar{f}(x_i)$$

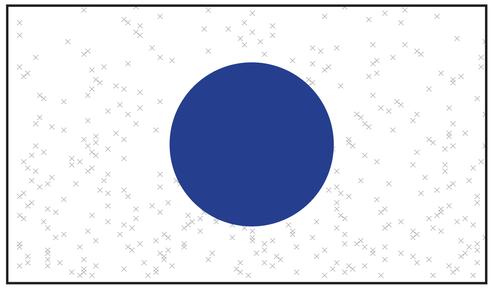
$$\bar{f}(x_i) = B\, \bar{f}_i$$

sampling theorem for sparse image representation



$$f(x) = \sum_{i=1}^{N} f_i\, \phi_i(x)$$

But what has this to do with sampling?

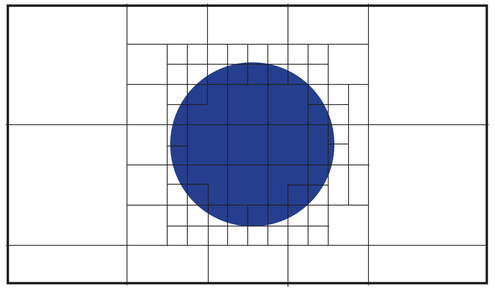


$$f(x) = \sum_{i=1}^{N} f(\lambda_i)\, \tilde{k}_i(x)$$

$$\bar{f}_i = B^{+}\, \bar{f}(x_i)$$

$$\bar{f}(x_i) = B\, \bar{f}_i$$

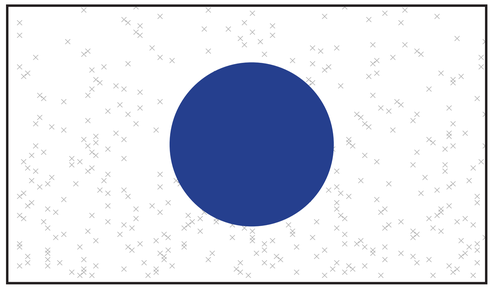
sampling theorem for sparse image representation



$$f(x) = \sum_{i=1}^{N} f_i\, \phi_i(x)$$

- optimal reconstruction kernels

But what has this to do with sampling?

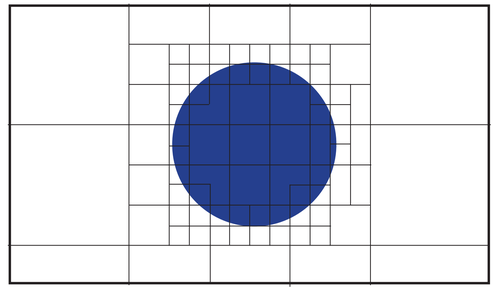


$$f(x) = \sum_{i=1}^{N} f(\lambda_i)\, \tilde{k}_i(x)$$

$$\bar{f}_i = B^+ \bar{f}(x_i)$$

$$\bar{f}(x_i) = B\, \bar{f}_i$$

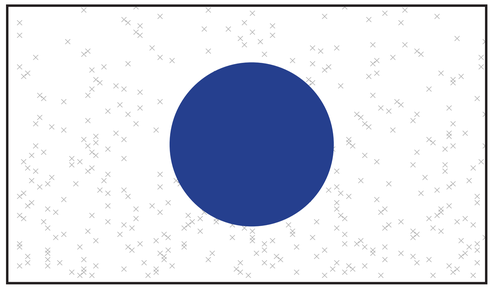
sampling theorem for sparse image representation



$$f(x) = \sum_{i=1}^{N} f_i\, \phi_i(x)$$

- optimal reconstruction kernels
- designed for non-bandlimited signals

But what has this to do with sampling?

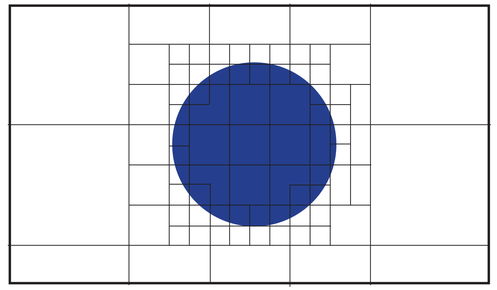


$$f(x) = \sum_{i=1}^{N} f(\lambda_i)\, \tilde{k}_i(x)$$

$$\bar{f}_i = B^+ \bar{f}(x_i)$$

$$\bar{f}(x_i) = B\, \bar{f}_i$$

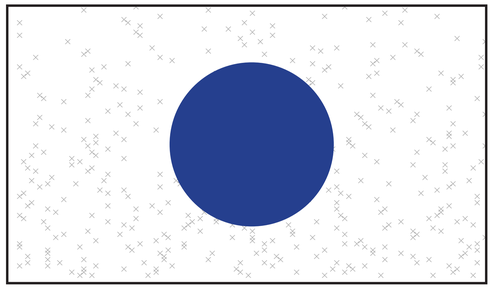
sampling theorem for sparse image representation



$$f(x) = \sum_{i=1}^{N} f_i\, \phi_i(x)$$

- optimal reconstruction kernels
- designed for non-bandlimited signals
- designed for non-uniform locations

But what has this to do with sampling?

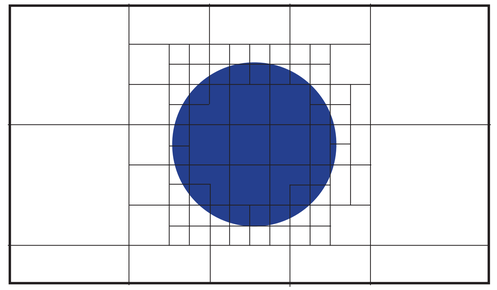


$$f(x) = \sum_{i=1}^{N} f(\lambda_i)\, \tilde{k}_i(x)$$

$$\bar{f}_i = B^+ \bar{f}(x_i)$$

$$\bar{f}(x_i) = B\, \bar{f}_i$$

sampling theorem for sparse image representation



$$f(x) = \sum_{i=1}^{N} f_i\, \phi_i(x)$$

- optimal reconstruction kernels
- designed for non-bandlimited signals
- designed for non-uniform locations
- allows to optimize sampling locations

Recipe

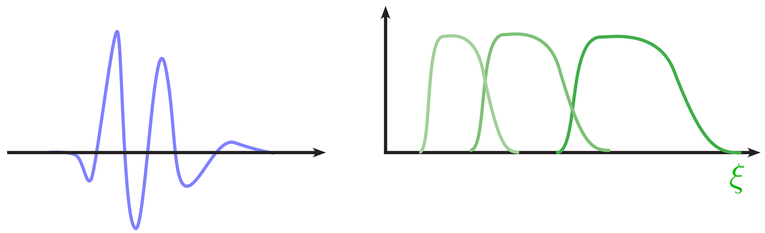
1. Find (approximate) tight function space
2. Sample function
3. Construct reproducing kernel basis

Recipe

1. Find (approximate) tight function space
2. Sample function
3. Construct reproducing kernel basis

And what's the right space?

$$f(x) = \sum_{i \in \mathcal{I}} f_i \boxed{\psi_i(x)} \approx \left(2^{-l}k, 2^{l}\xi\right)$$



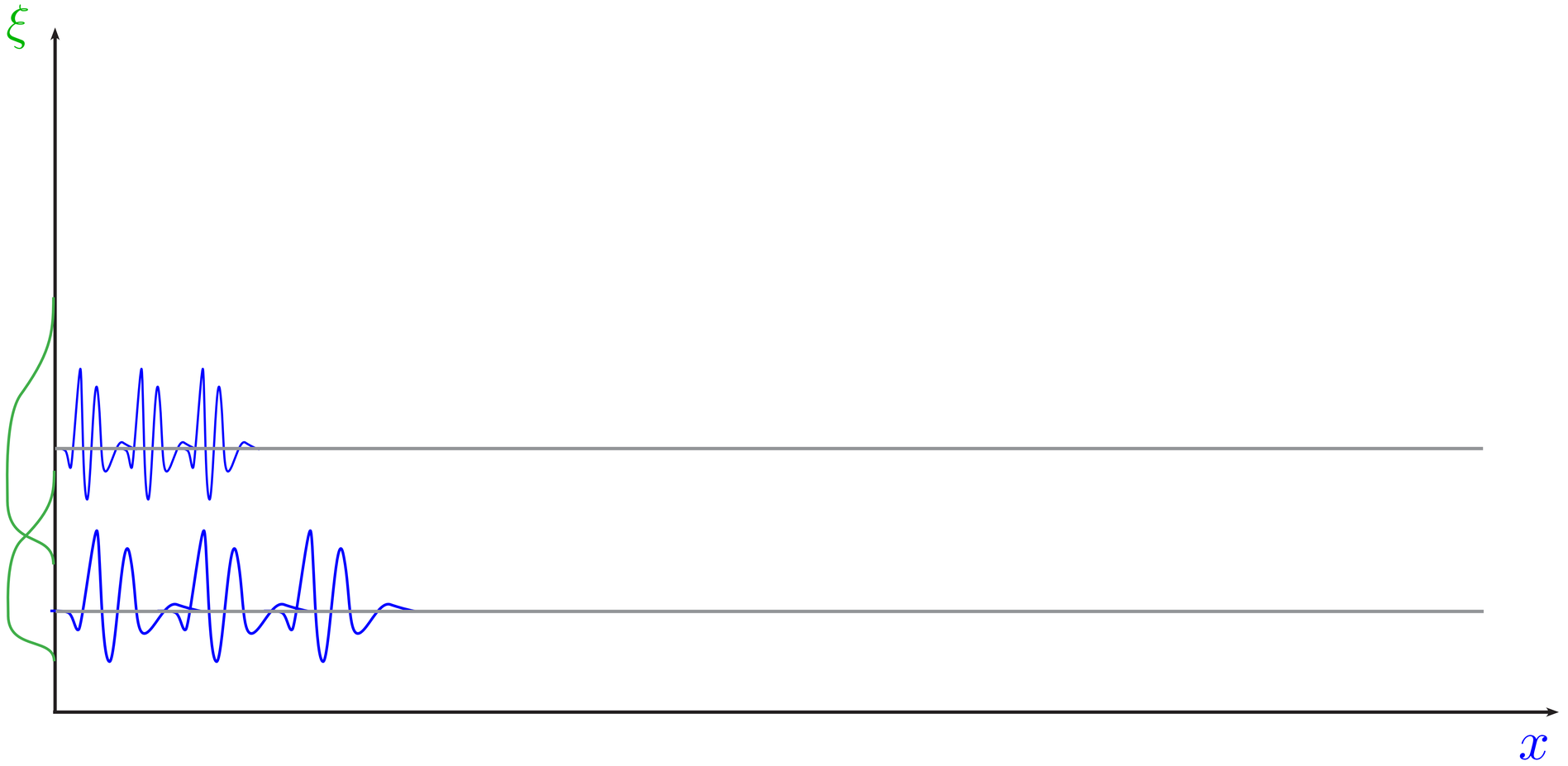
And what's the right space?

And what's the right space?

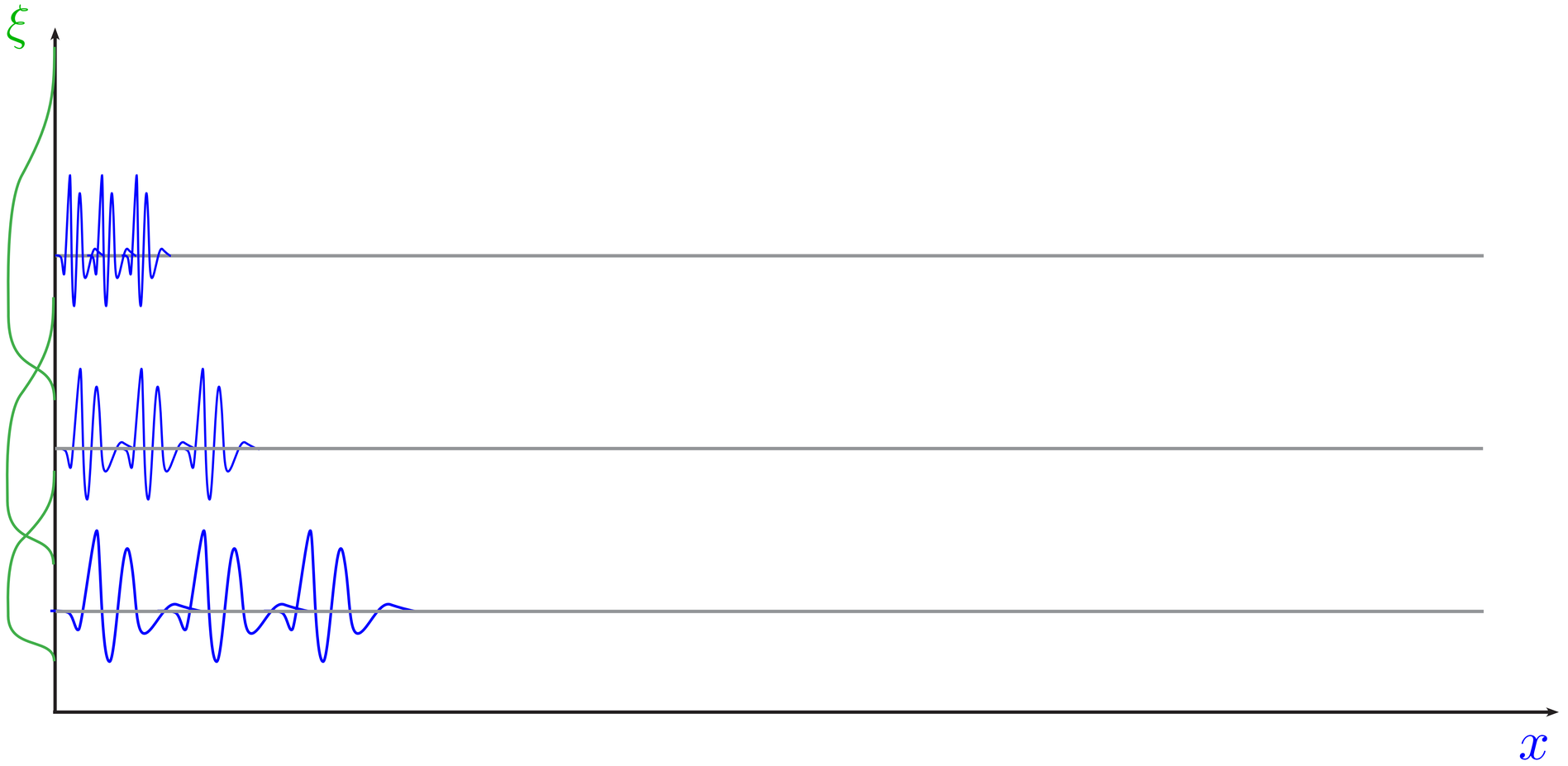
And what's the right space?

And what's the right space?

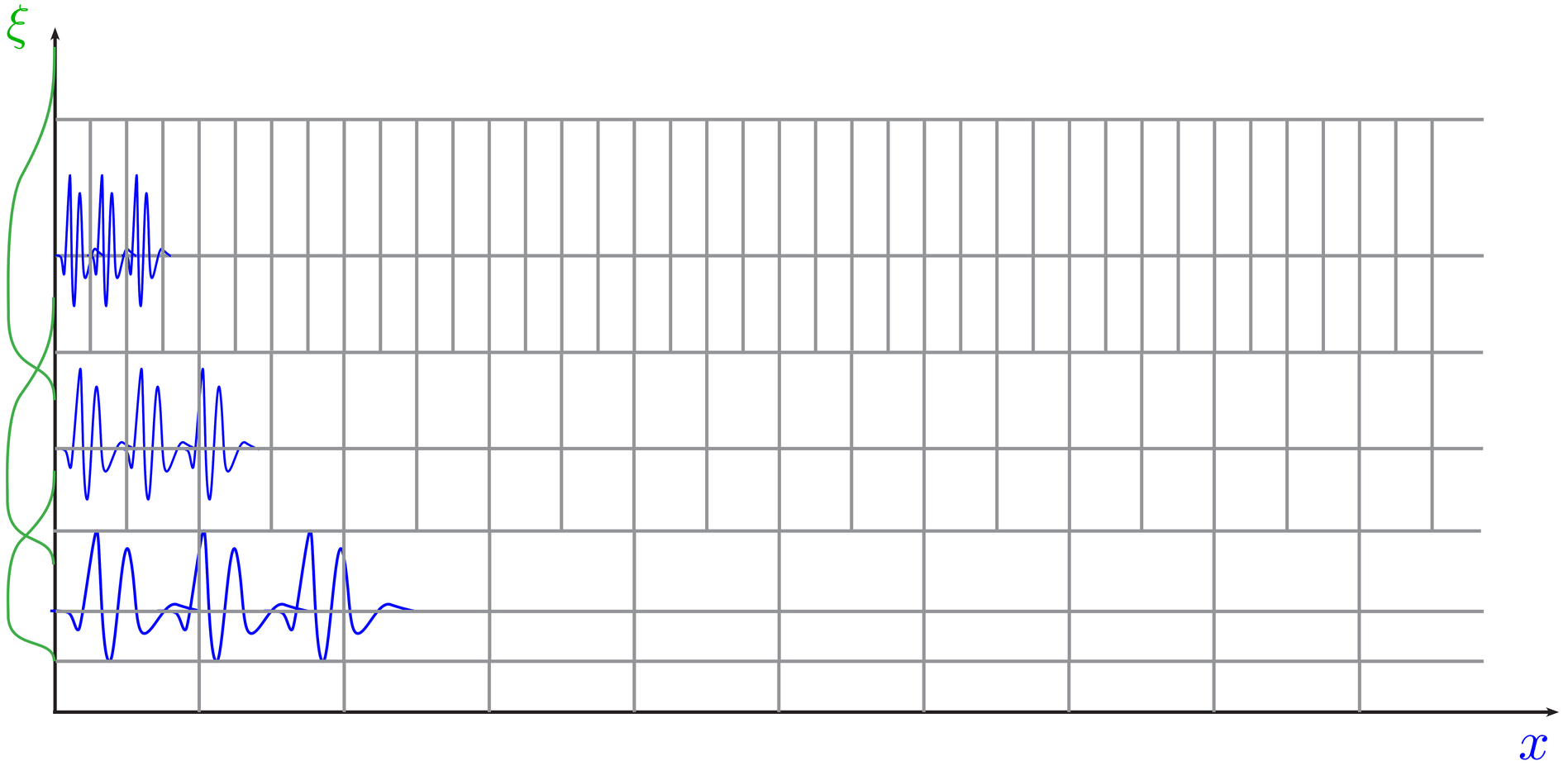
And what's the right space?



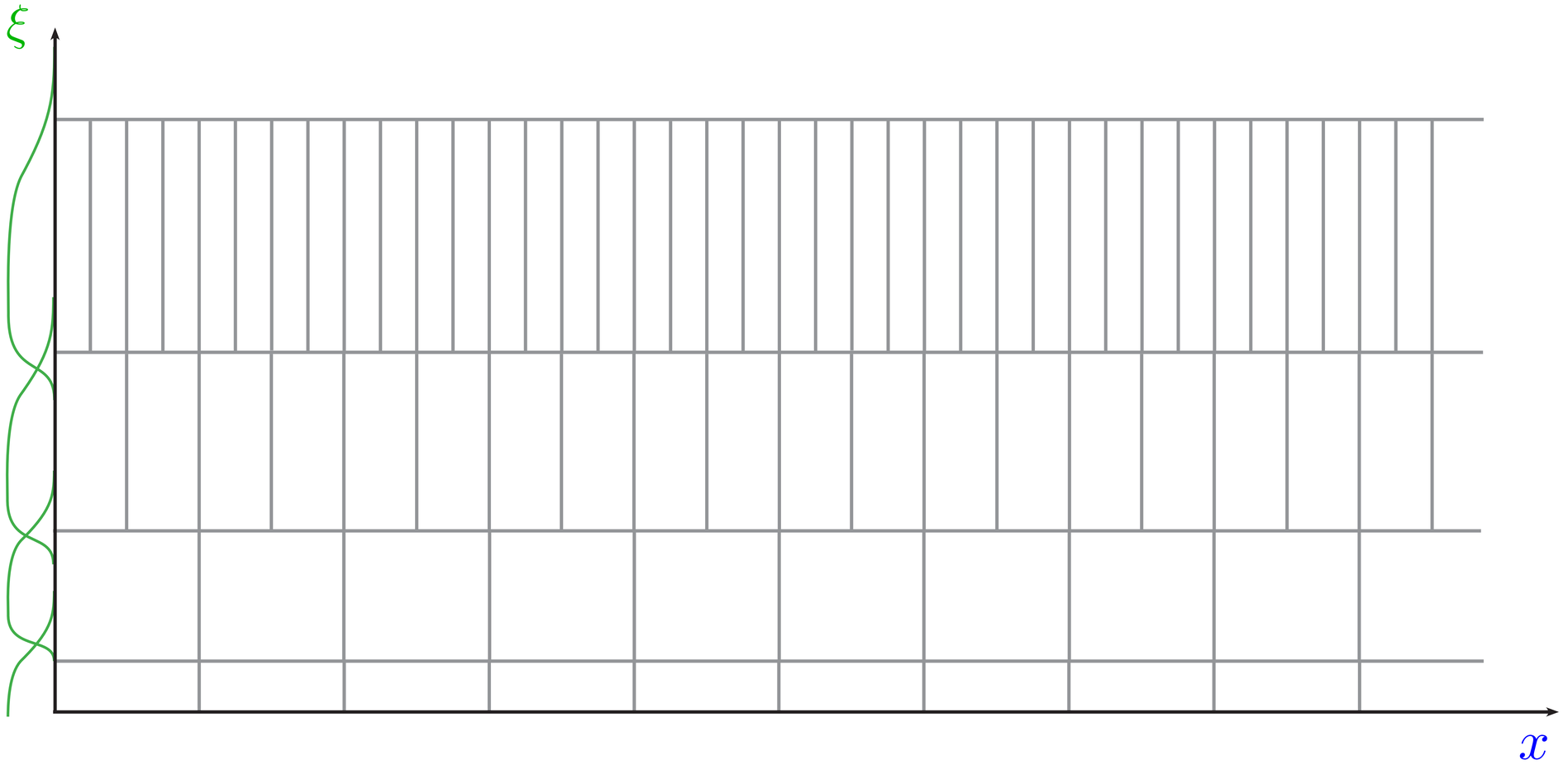
And what's the right space?



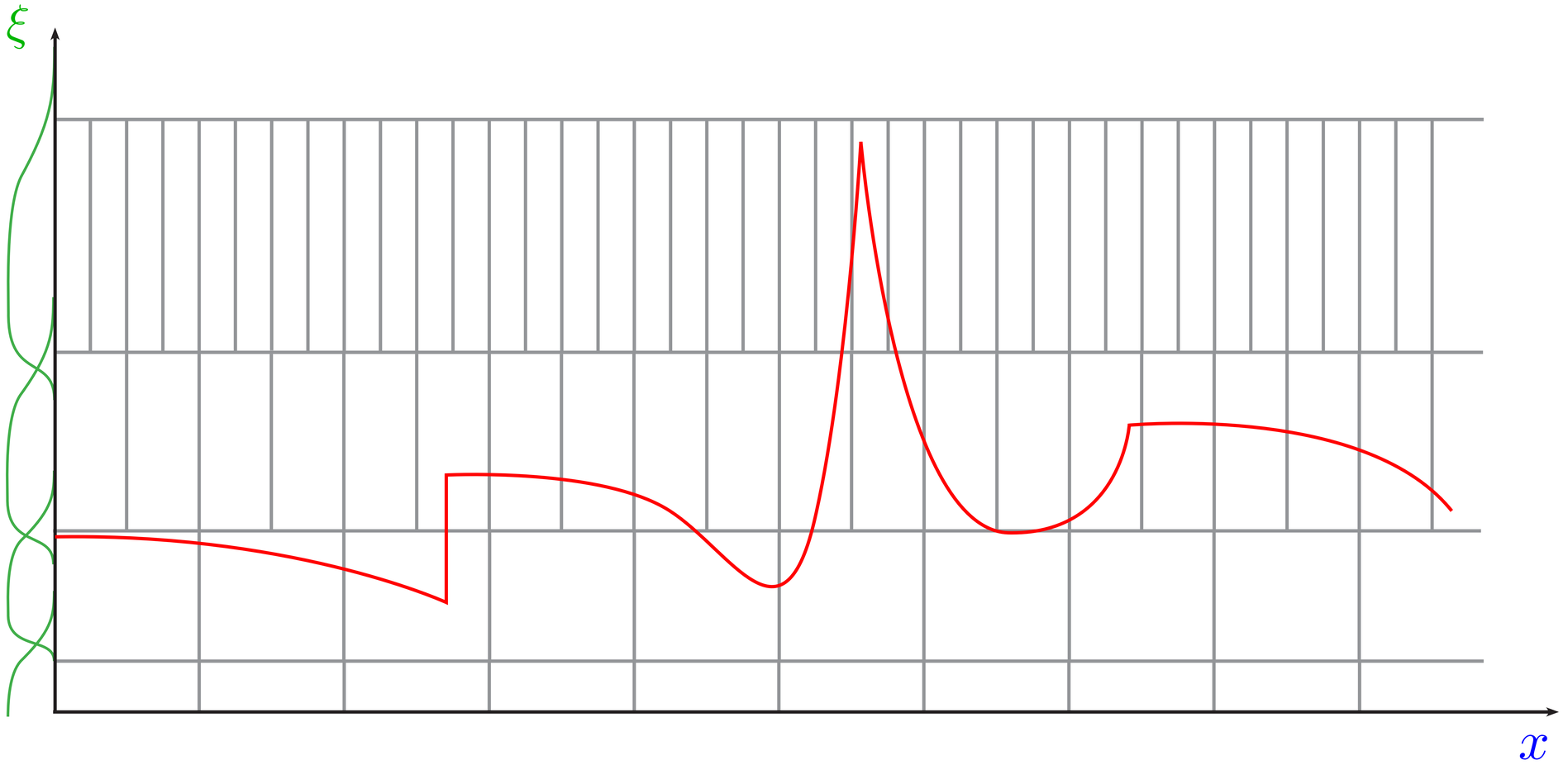
And what's the right space?



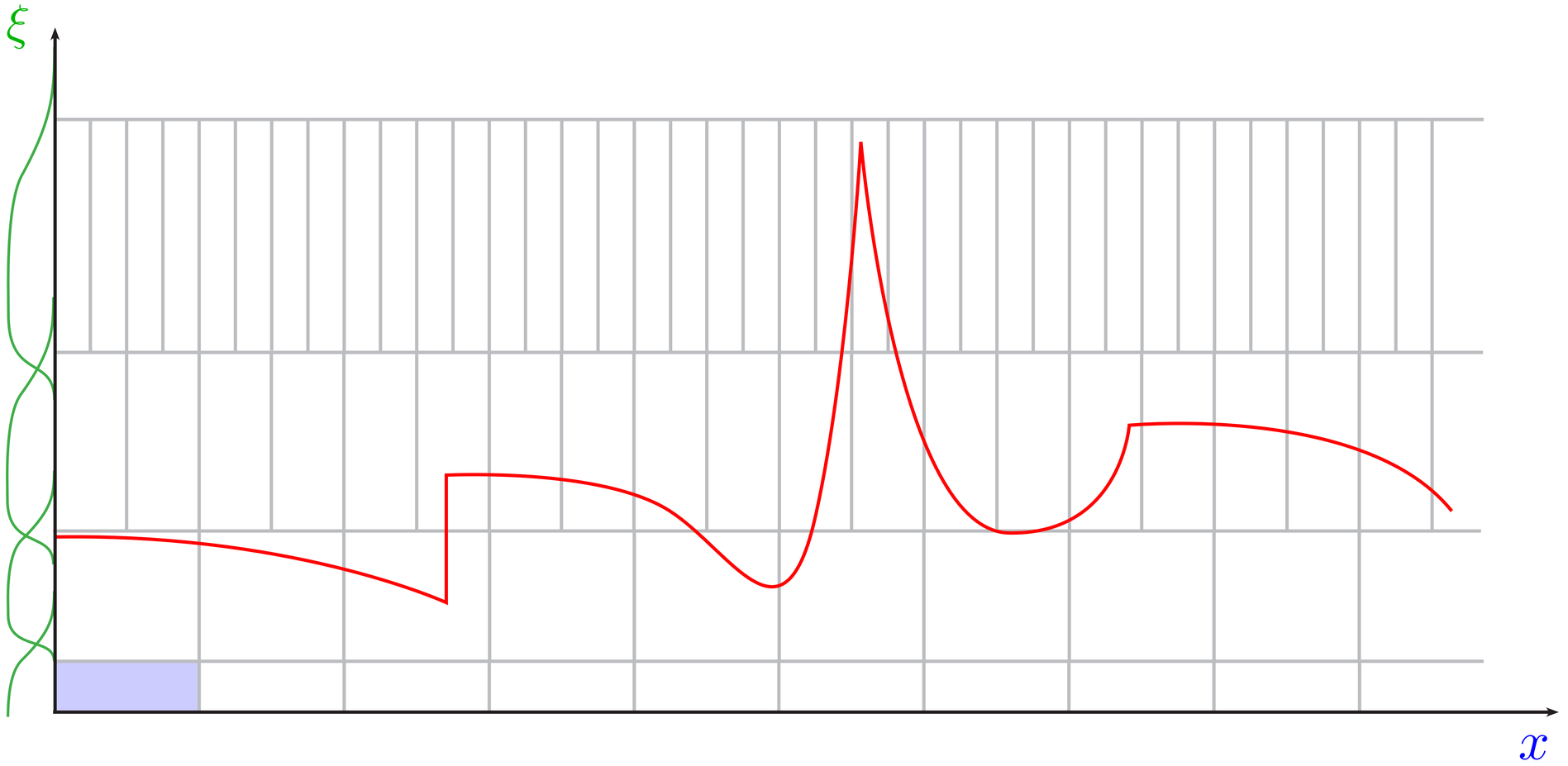
And what's the right space?



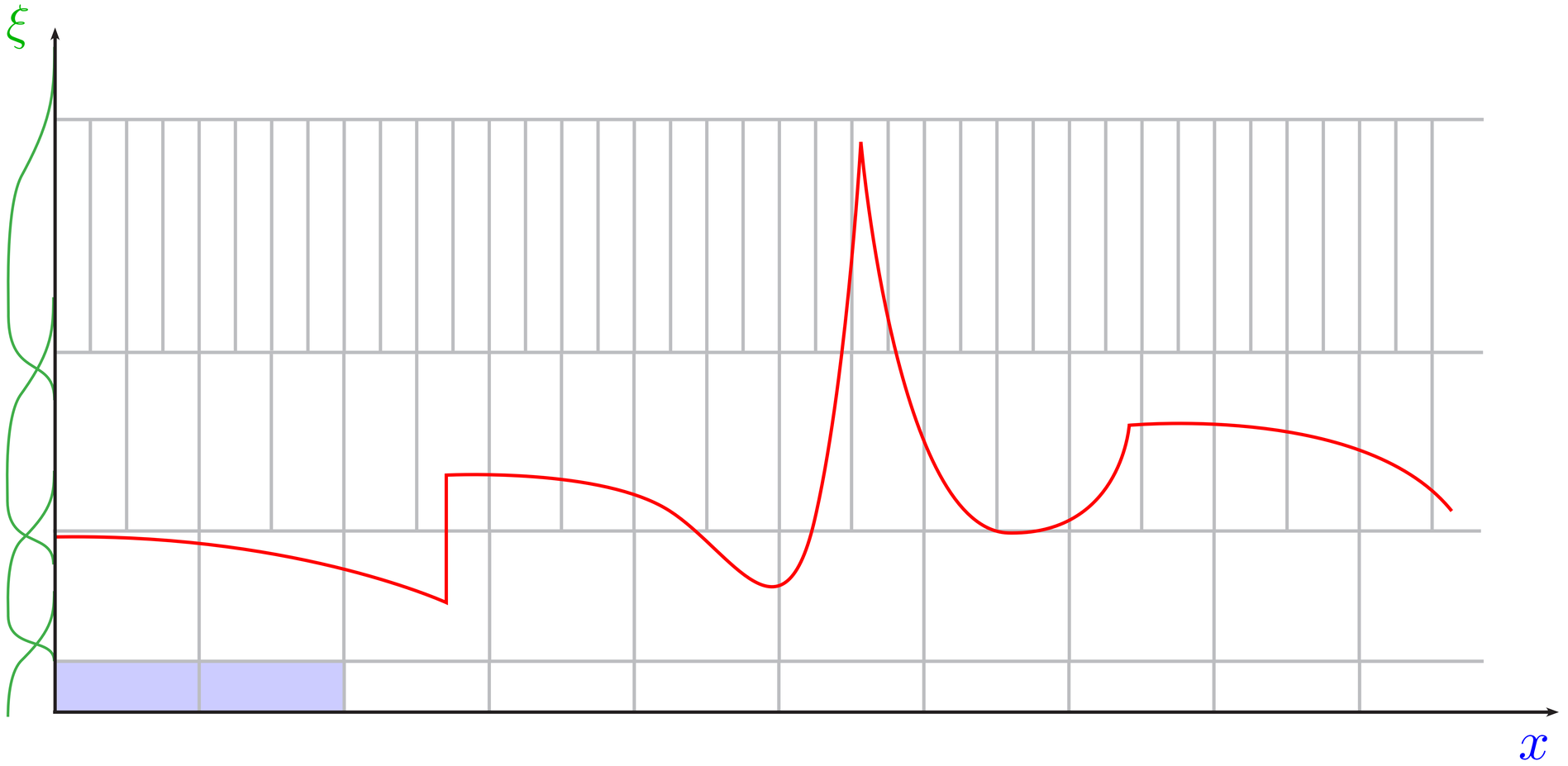
And what's the right space?



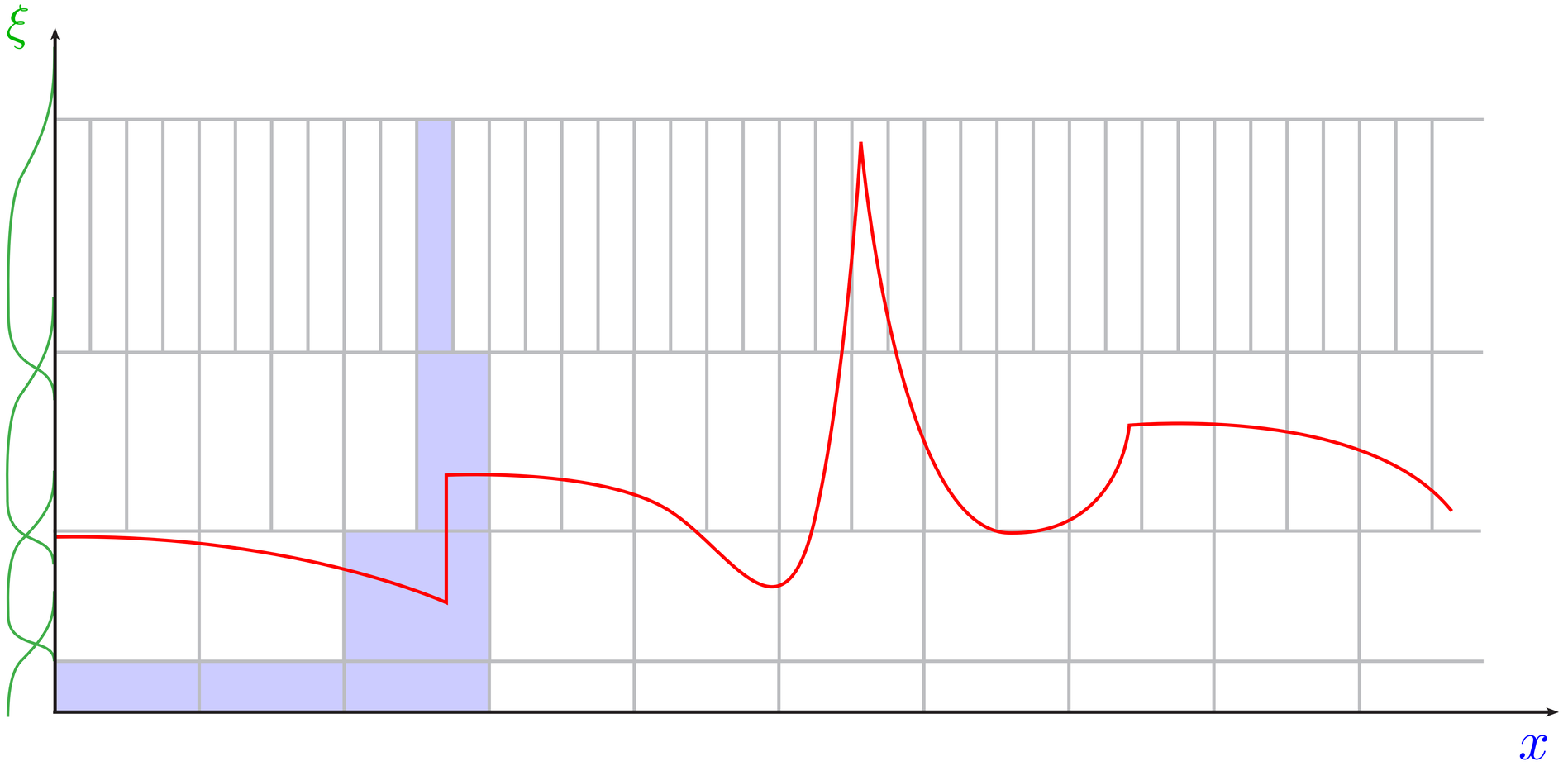
And what's the right space?



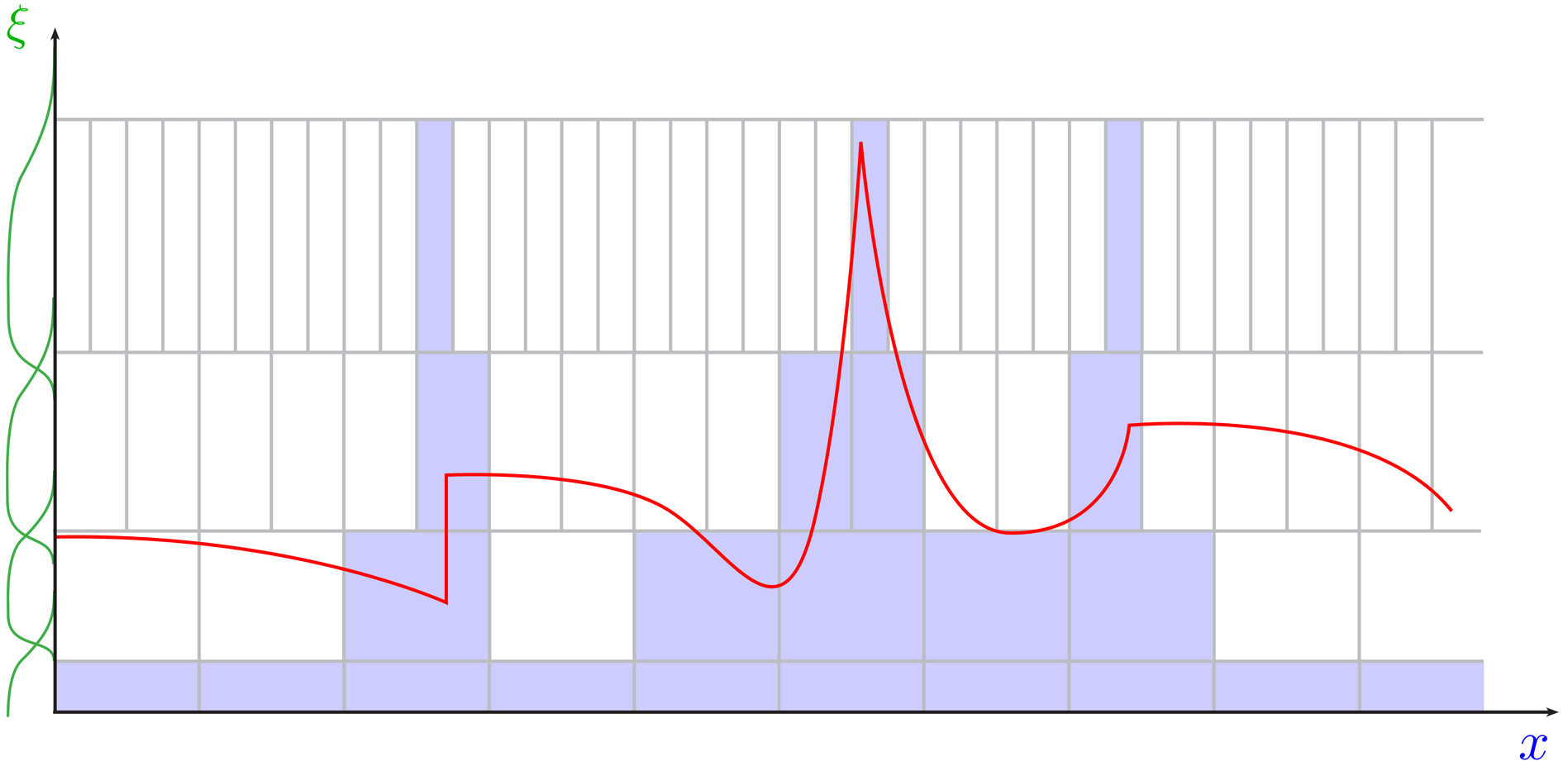
And what's the right space?



And what's the right space?



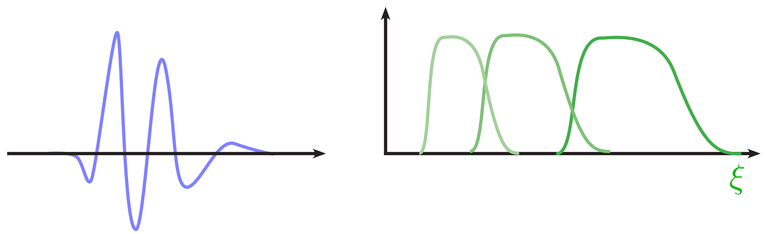
And what's the right space?



And what's the right space?

And what's the right space?

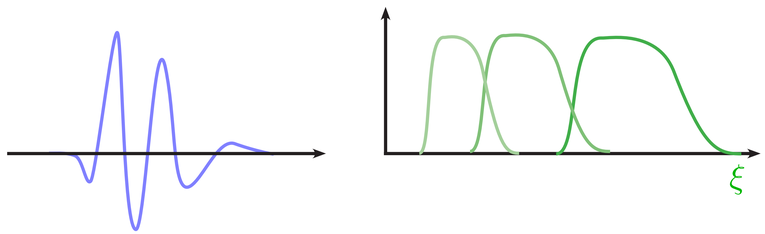
$$f(x) = \sum_{i \in \mathcal{I}} f_i \boxed{\psi_i(x)} \approx \left(2^{-l}k, 2^l \xi\right)$$



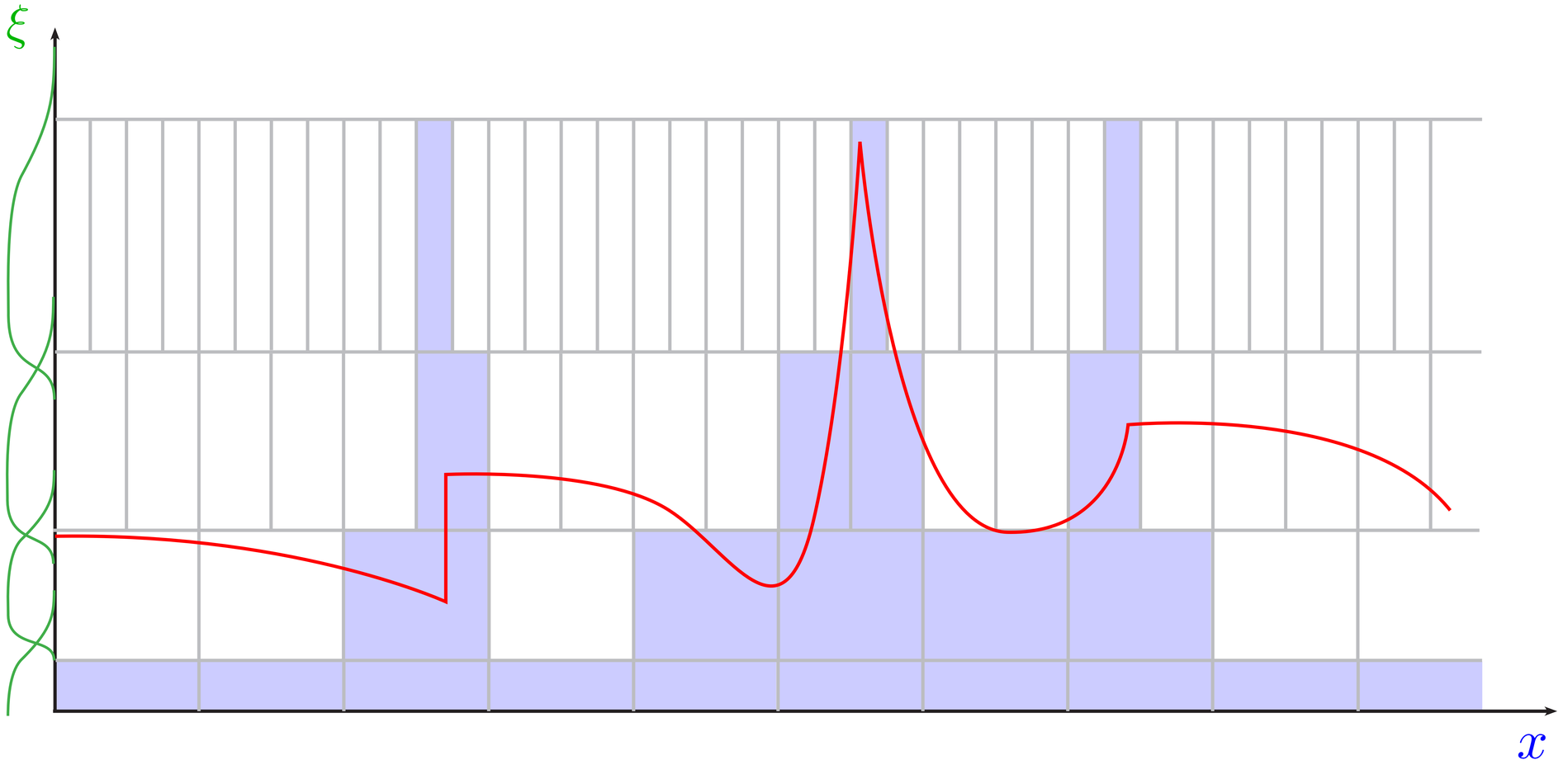
And what's the right space?

$$f(x) = \sum_{i \in \mathcal{I}} f_i \boxed{\psi_i(x)} \approx \left(2^{-l}k, 2^{l}\xi\right)$$

depends on image function



And what's the right space?



And what's the right space?

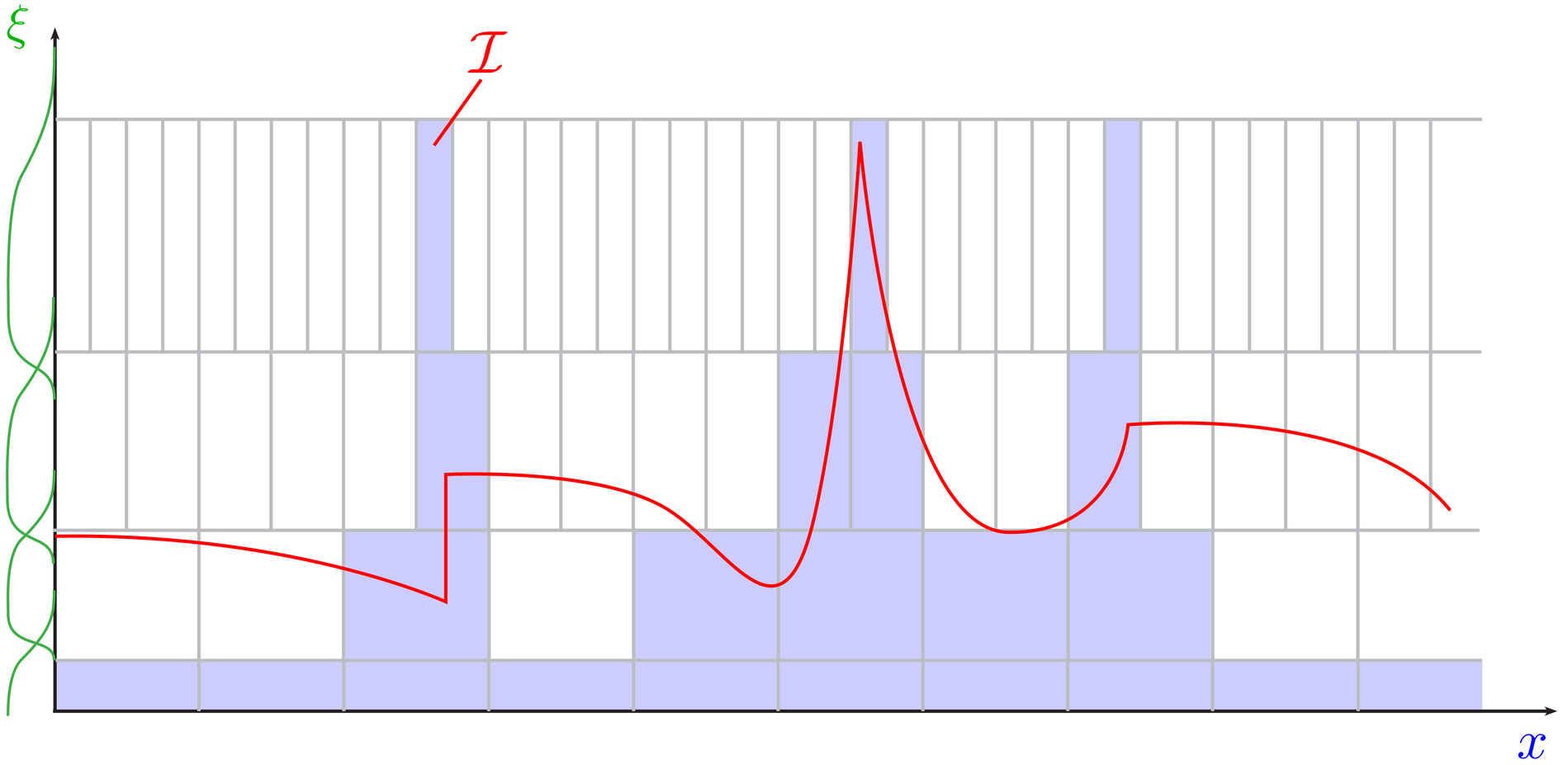
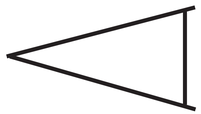


Image generation



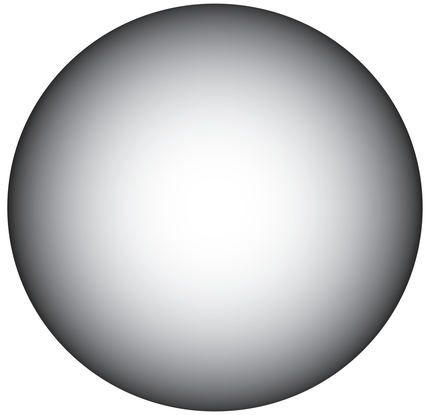
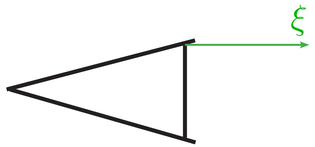


Image generation



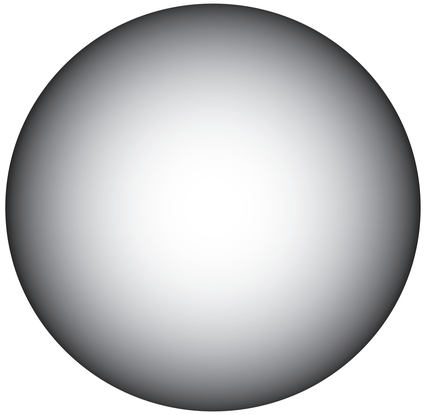


Image generation

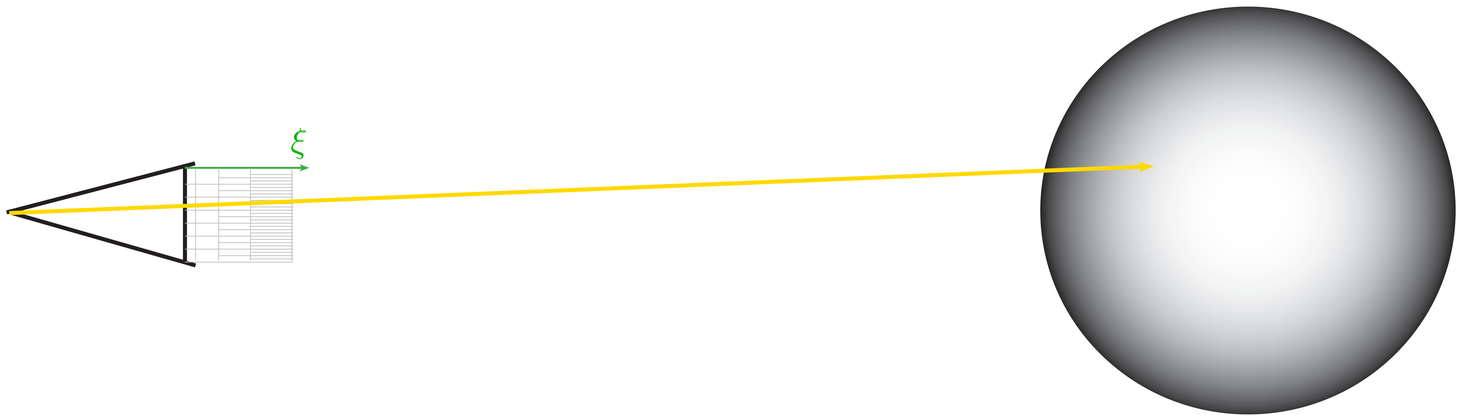


Image generation

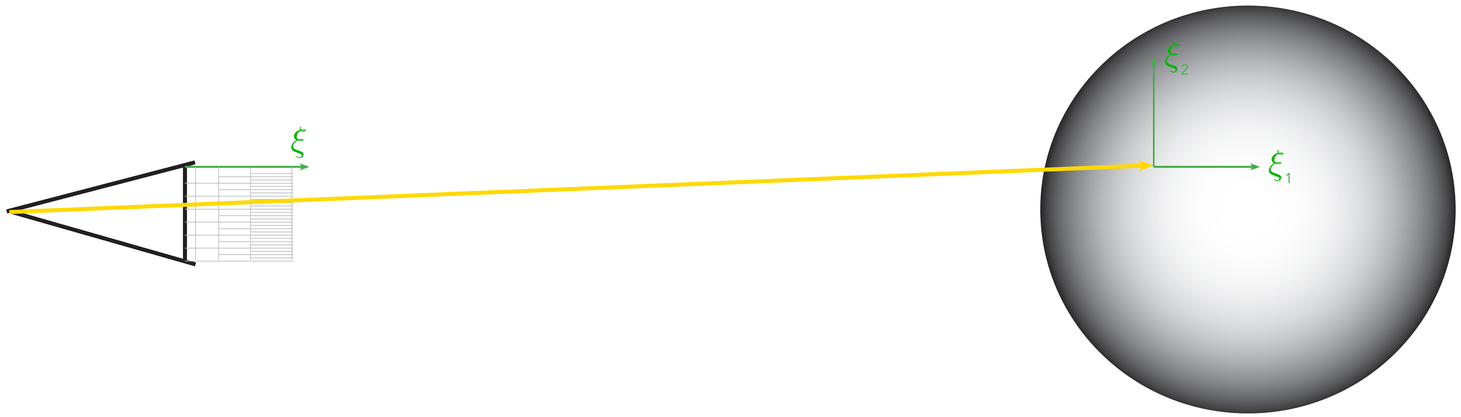


Image generation

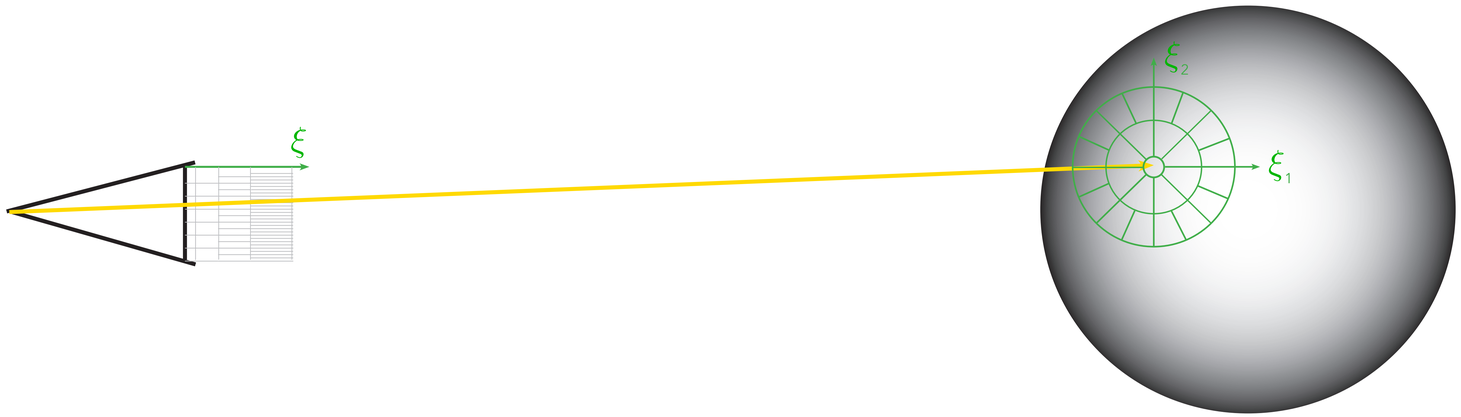


Image generation

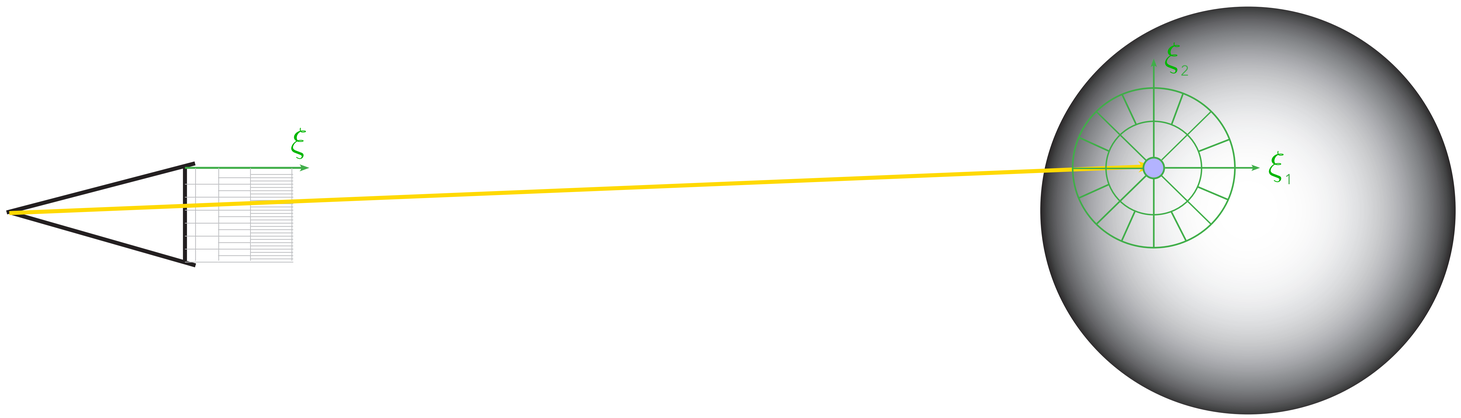


Image generation

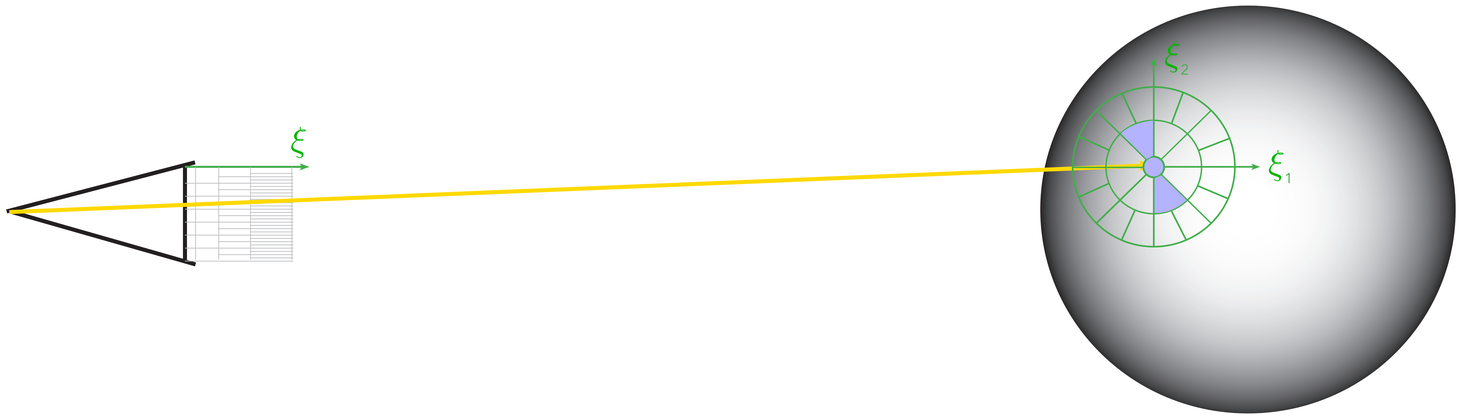


Image generation

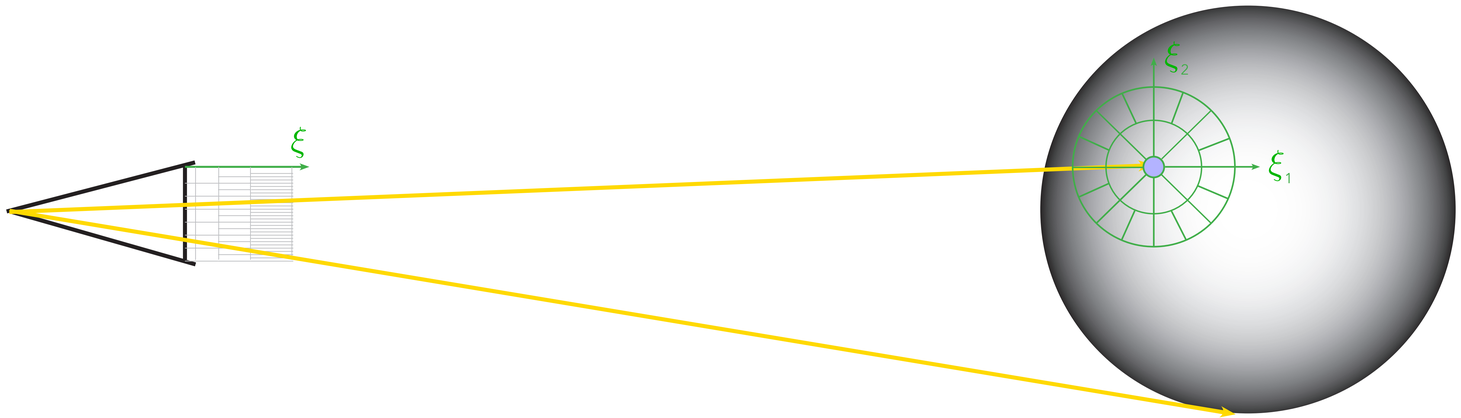


Image generation

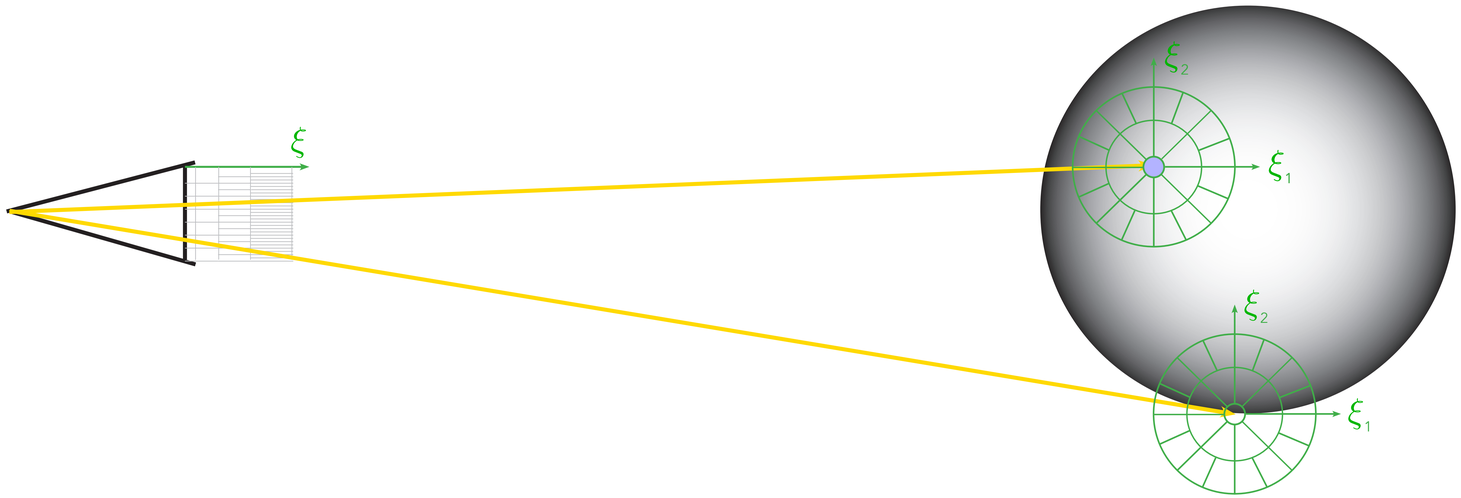


Image generation

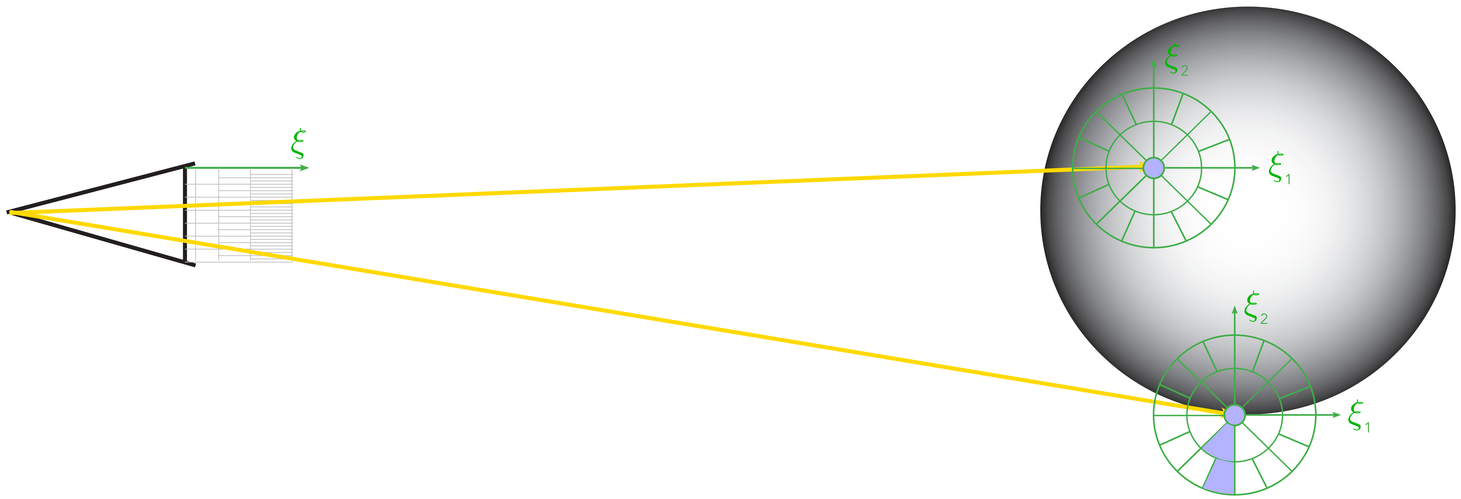


Image generation

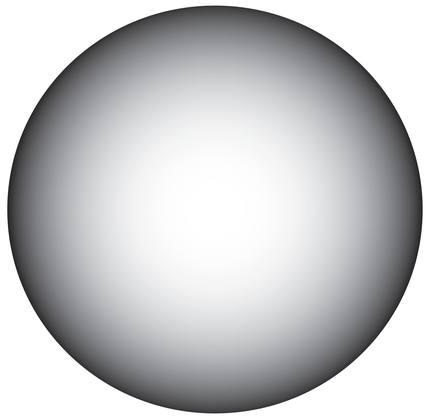
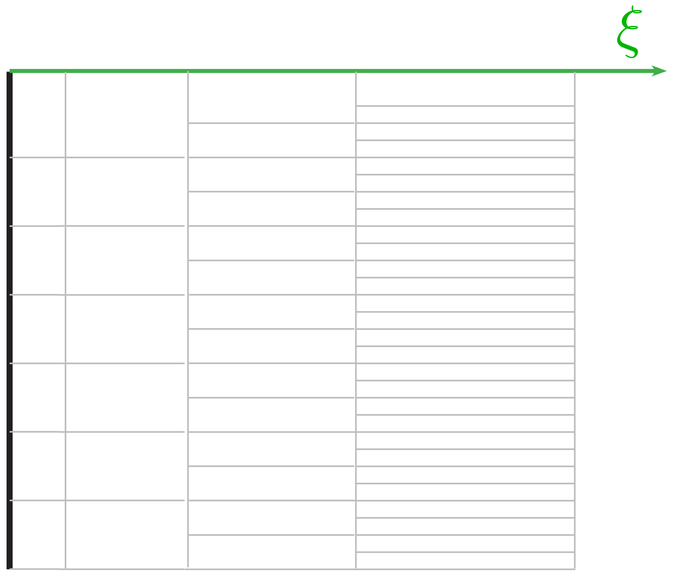


Image generation



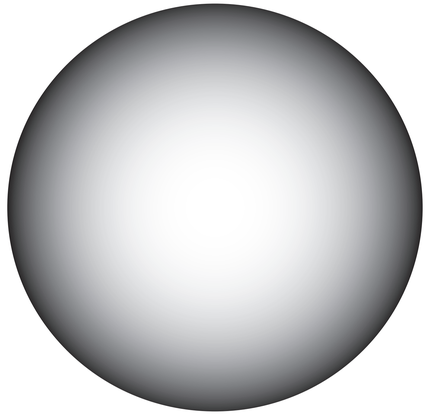


Image generation

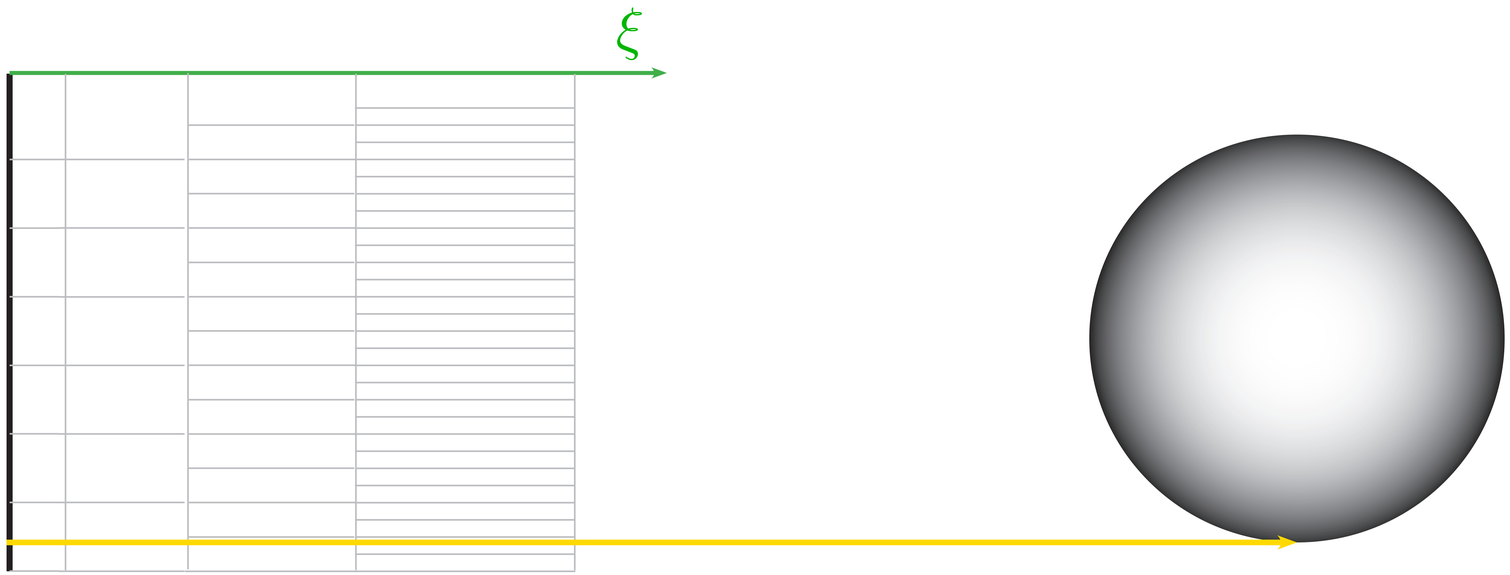


Image generation

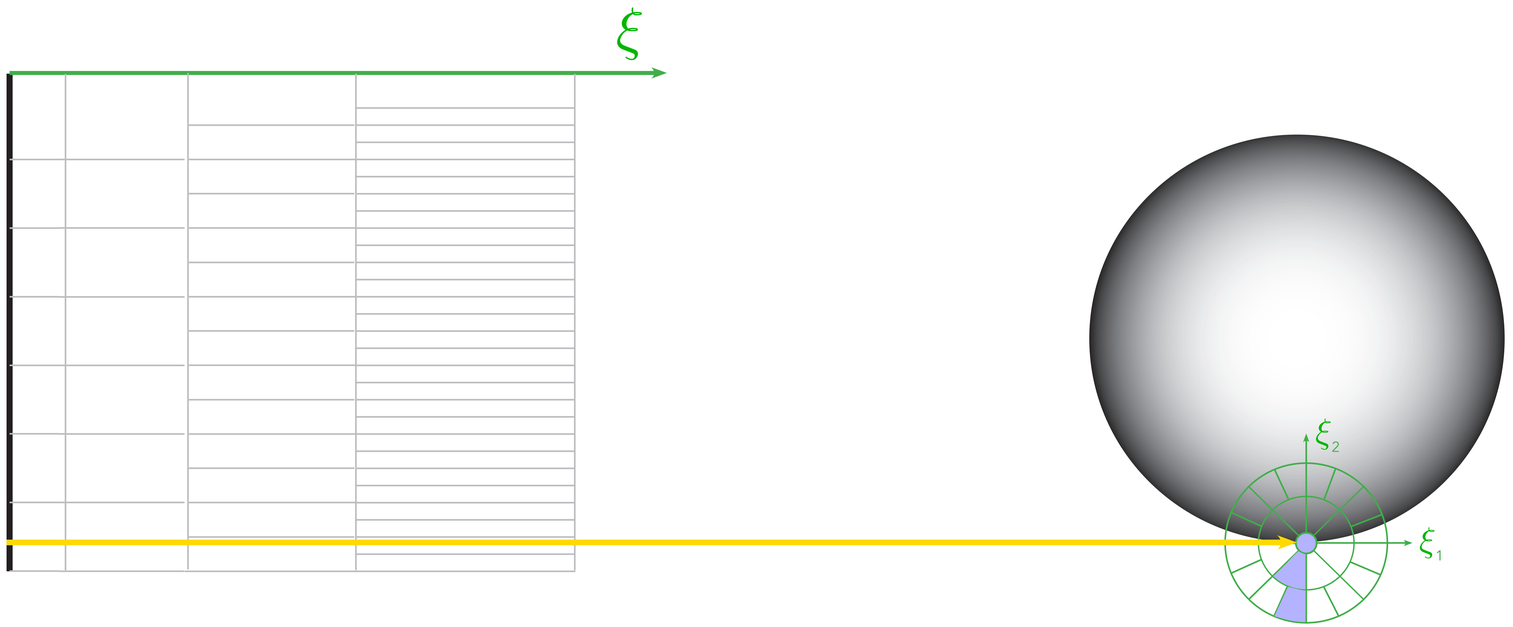


Image generation

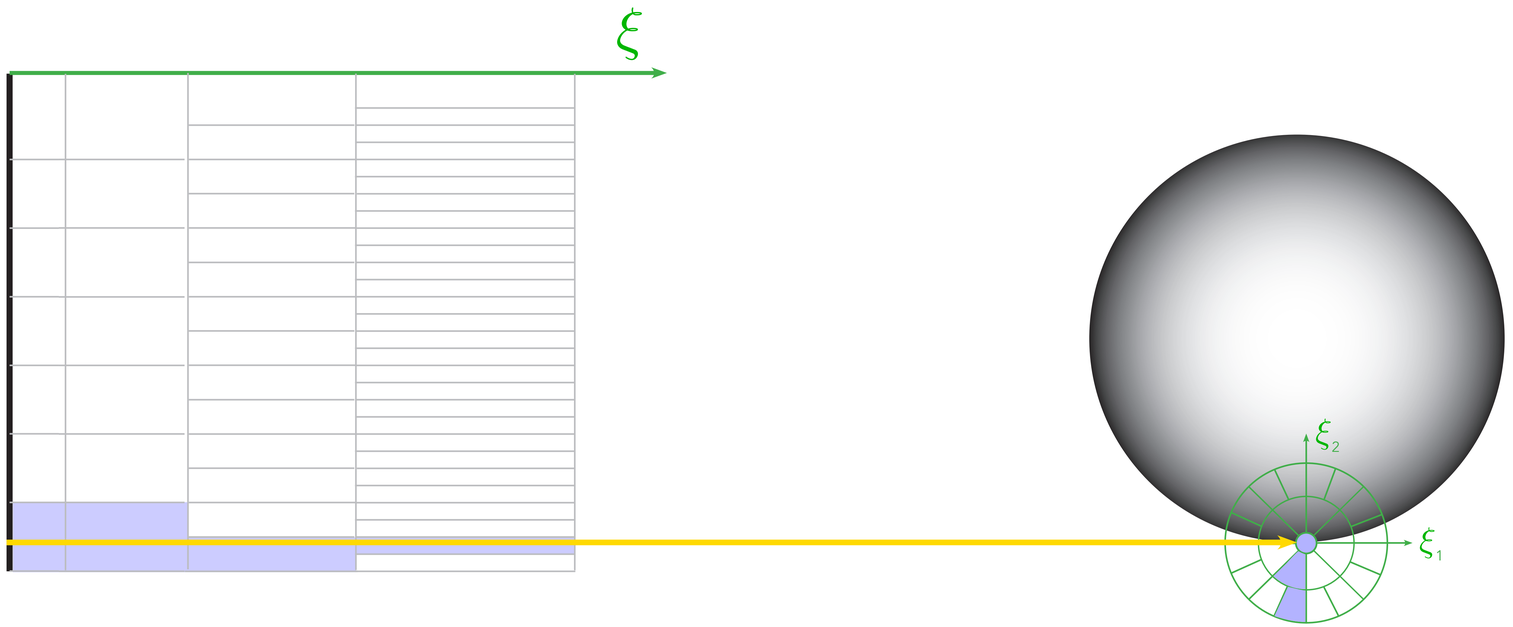


Image generation

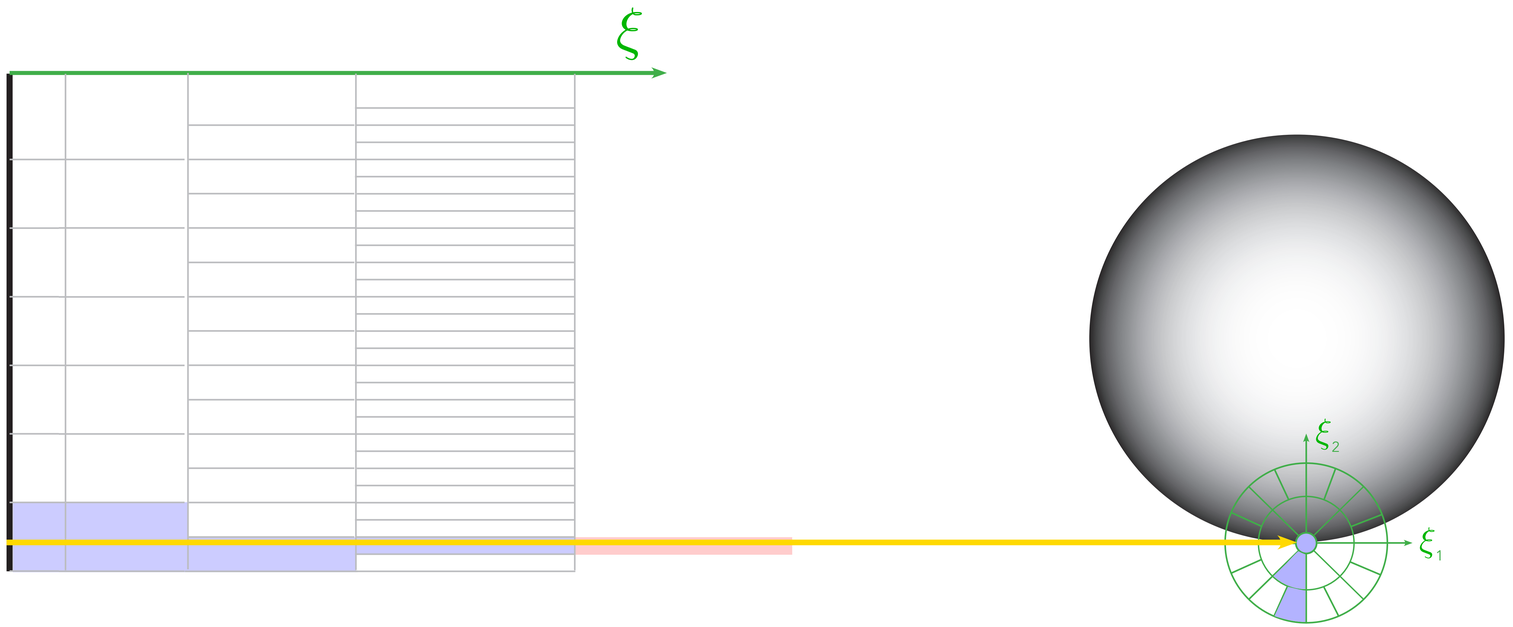


Image generation

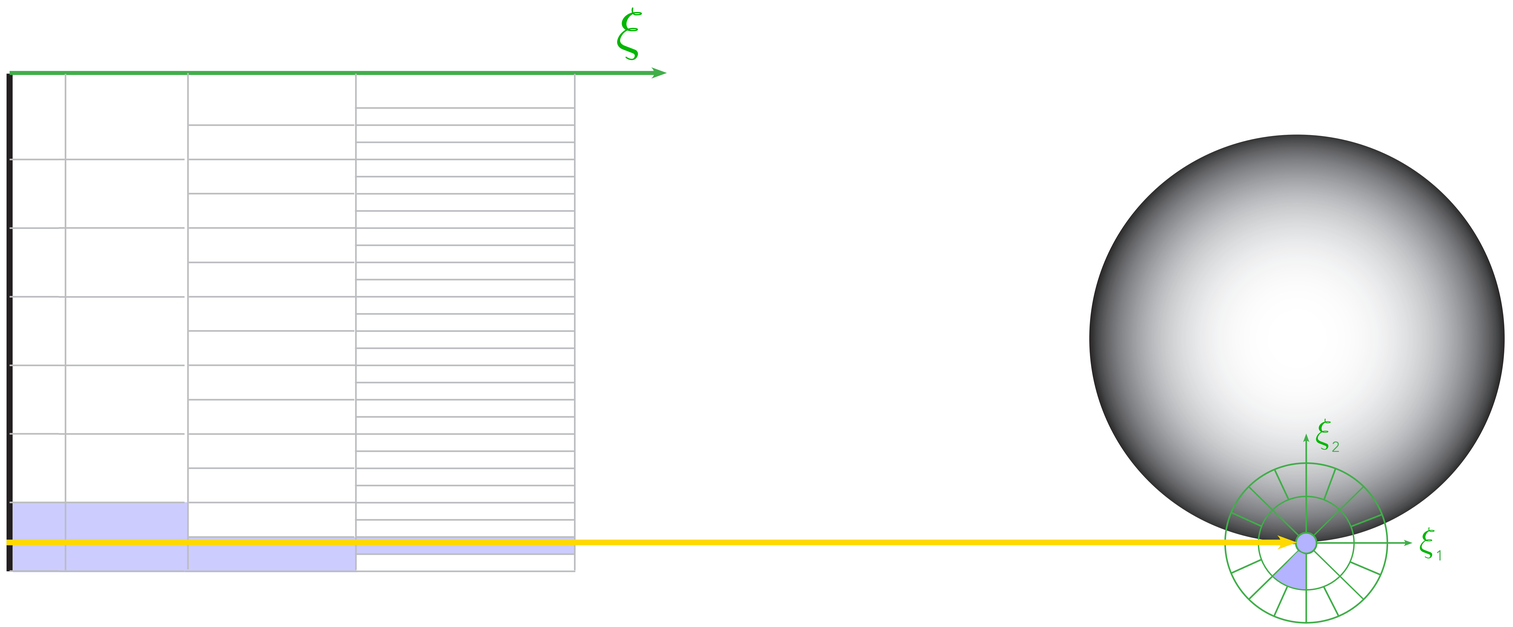


Image generation

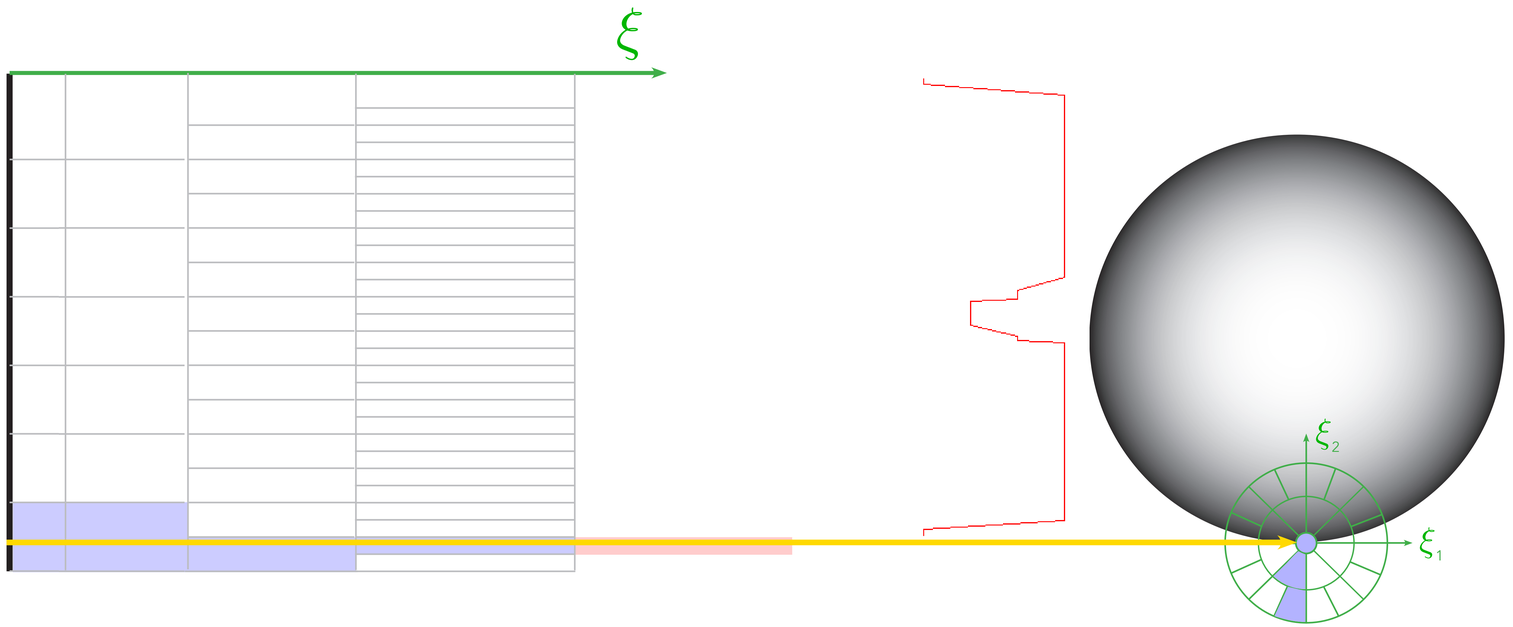


Image generation

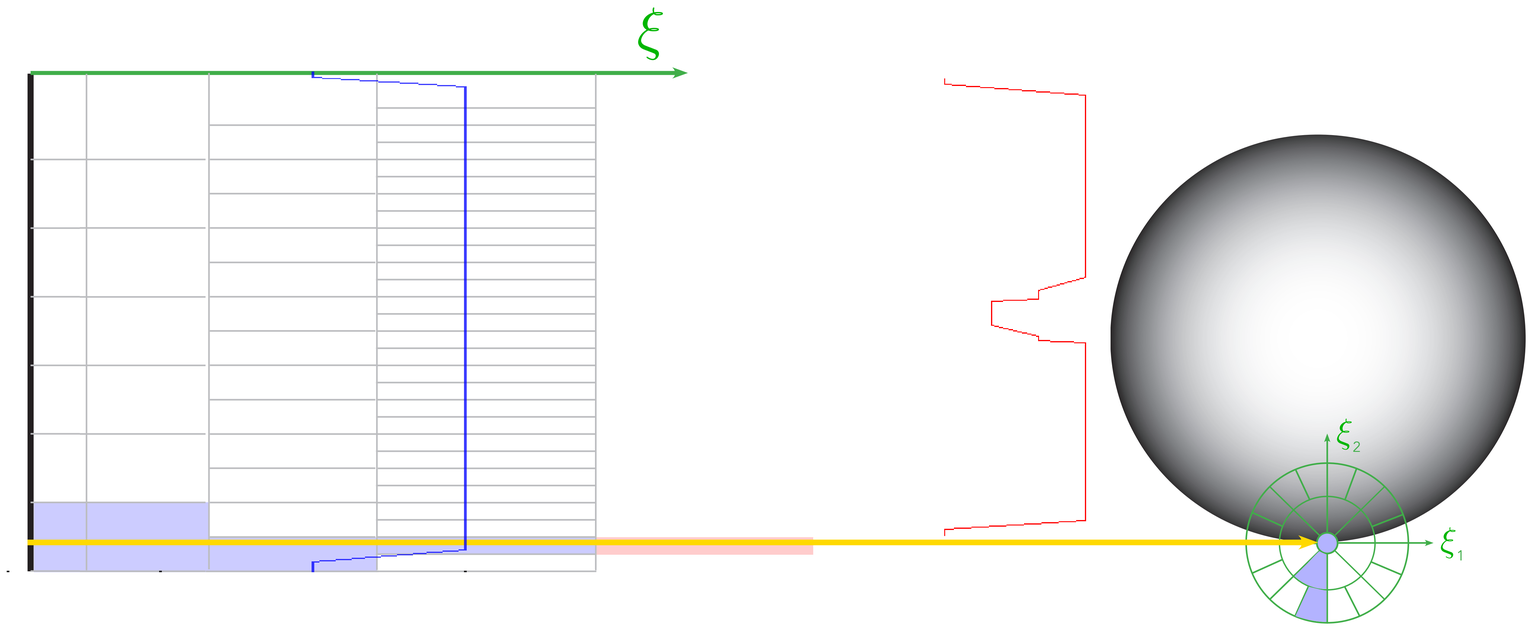


Image generation

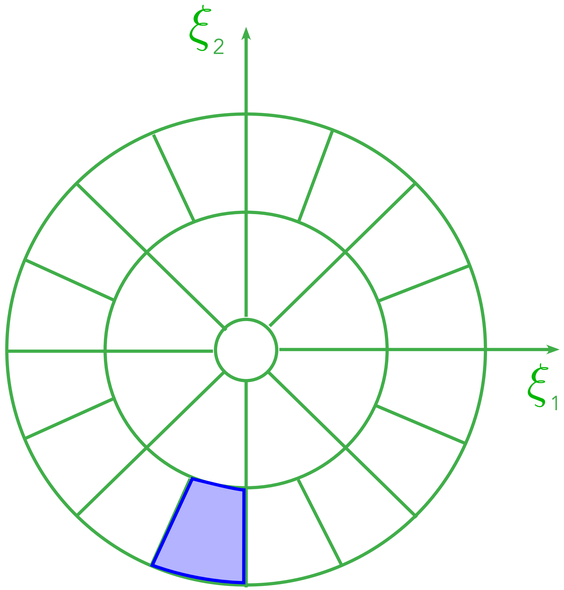
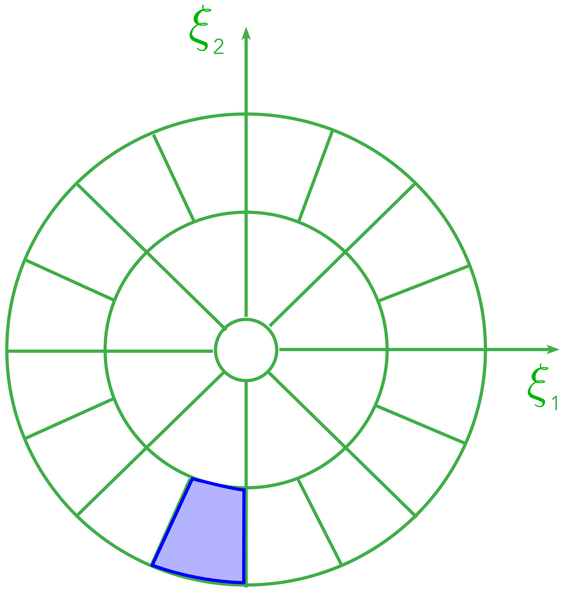


Image generation



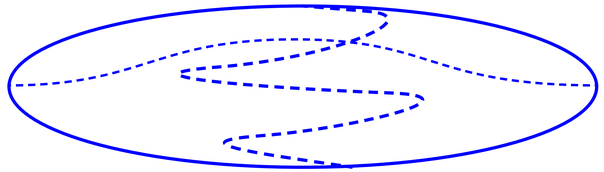
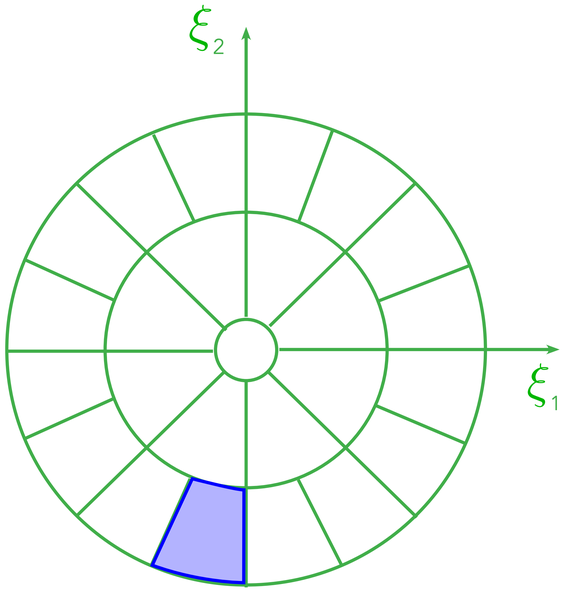
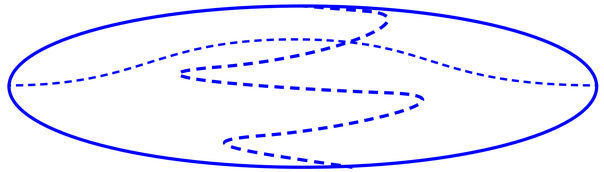


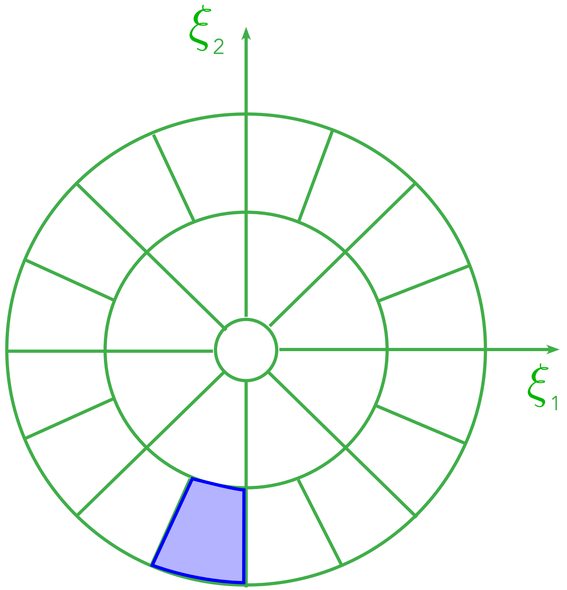
Image generation

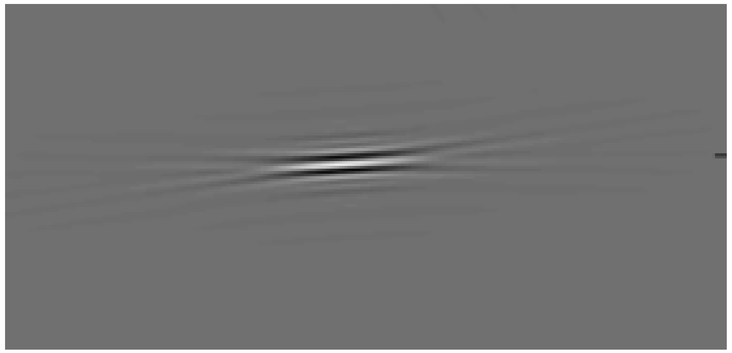




curvelet / shearlet / contourlet

Image generation





curvelet / shearlet / contourlet

Image generation

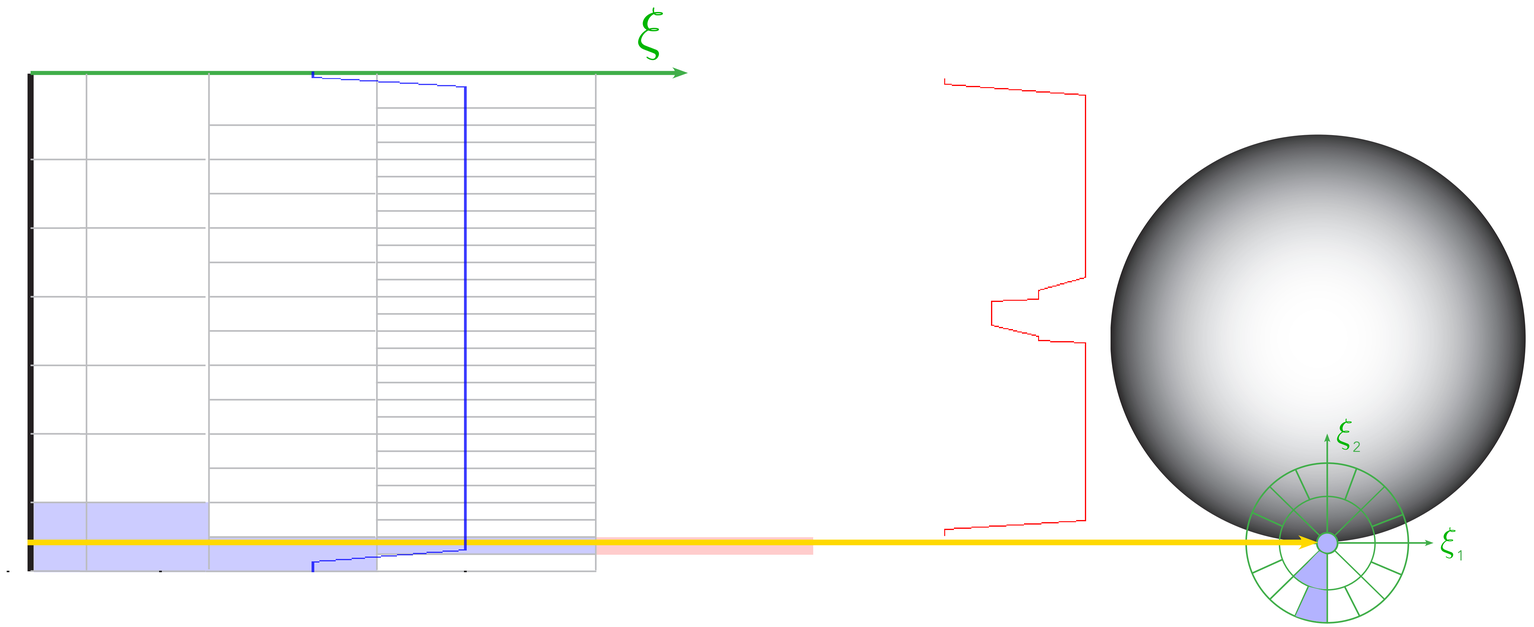


Image generation

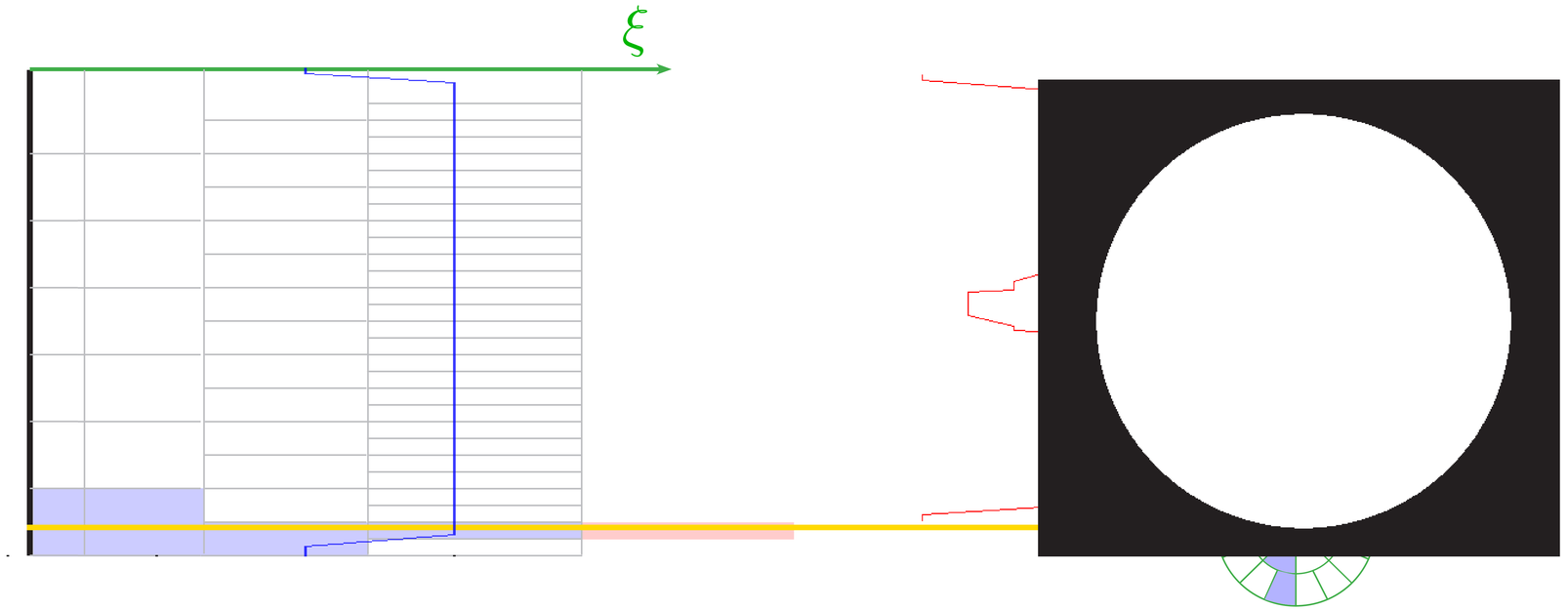


Image generation

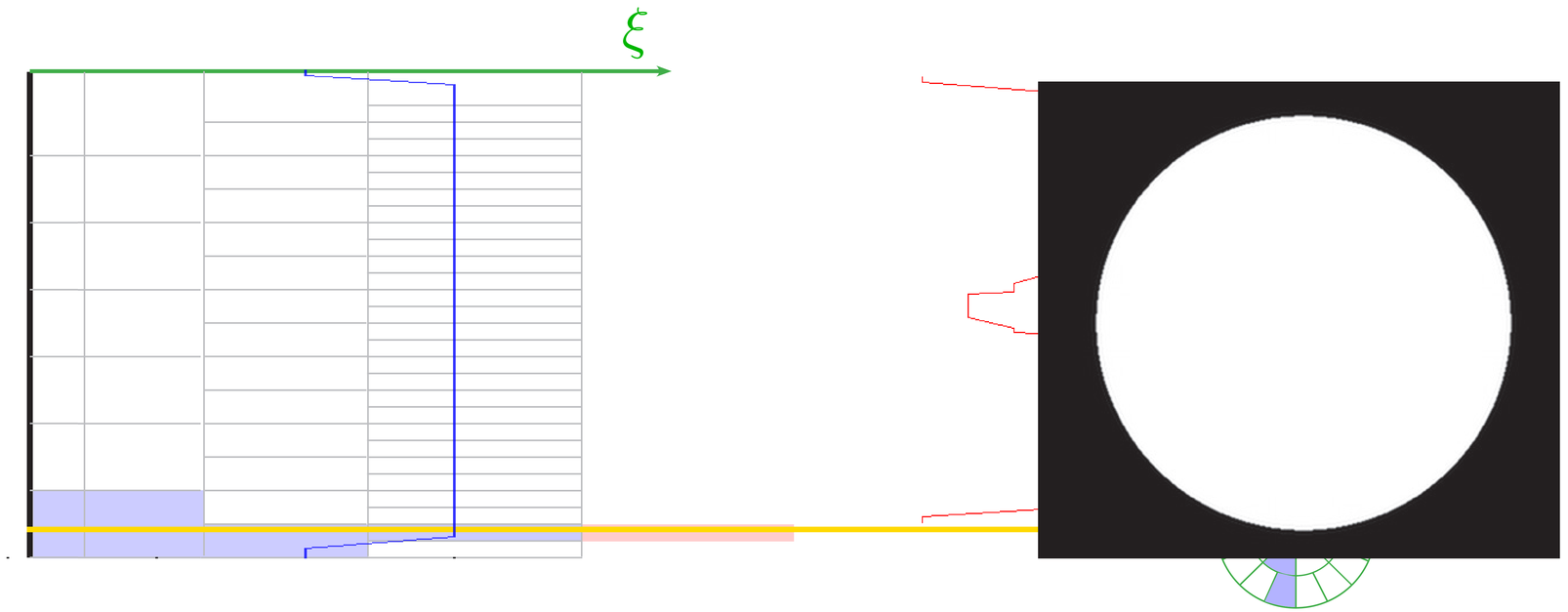


Image generation

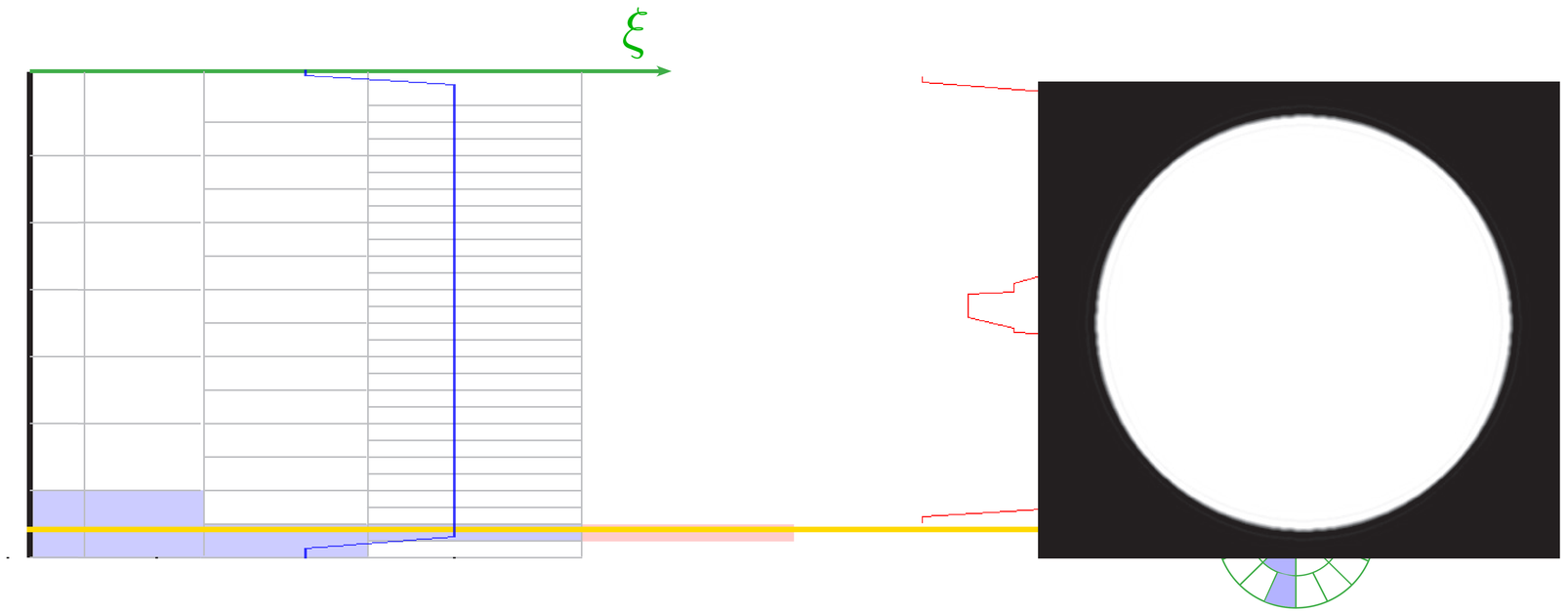


Image generation

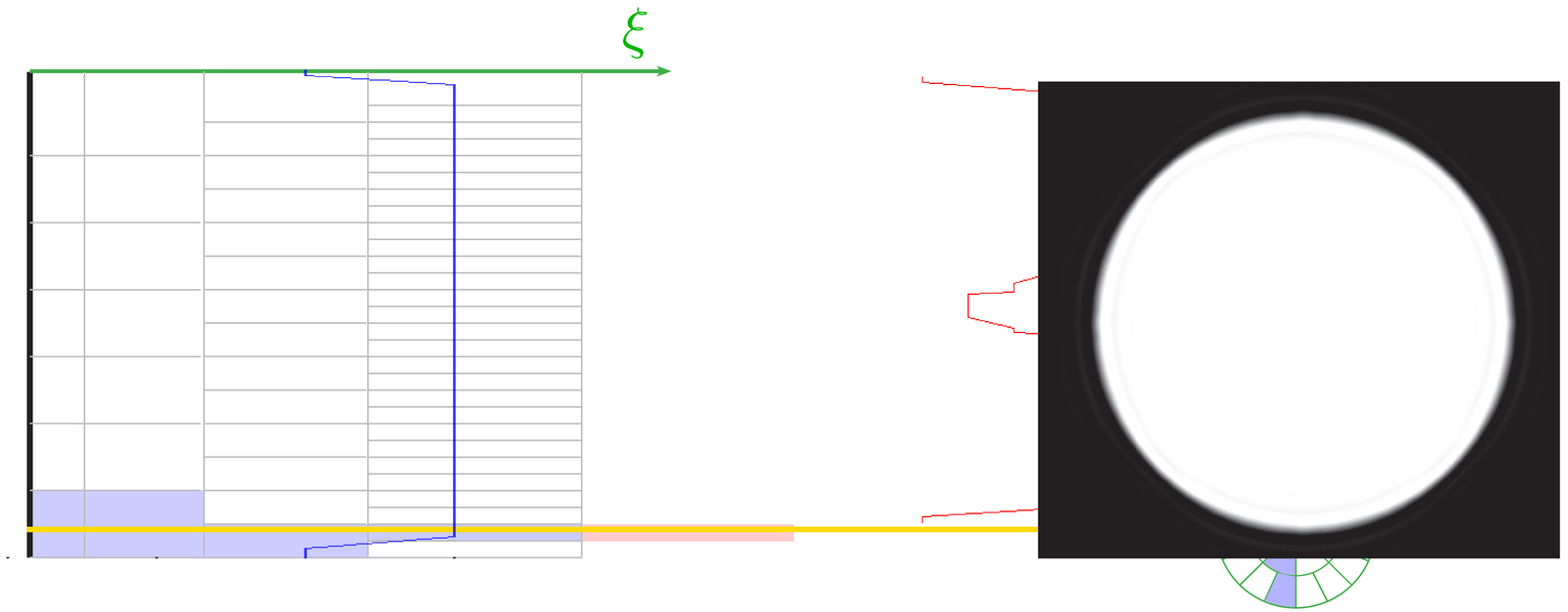


Image generation

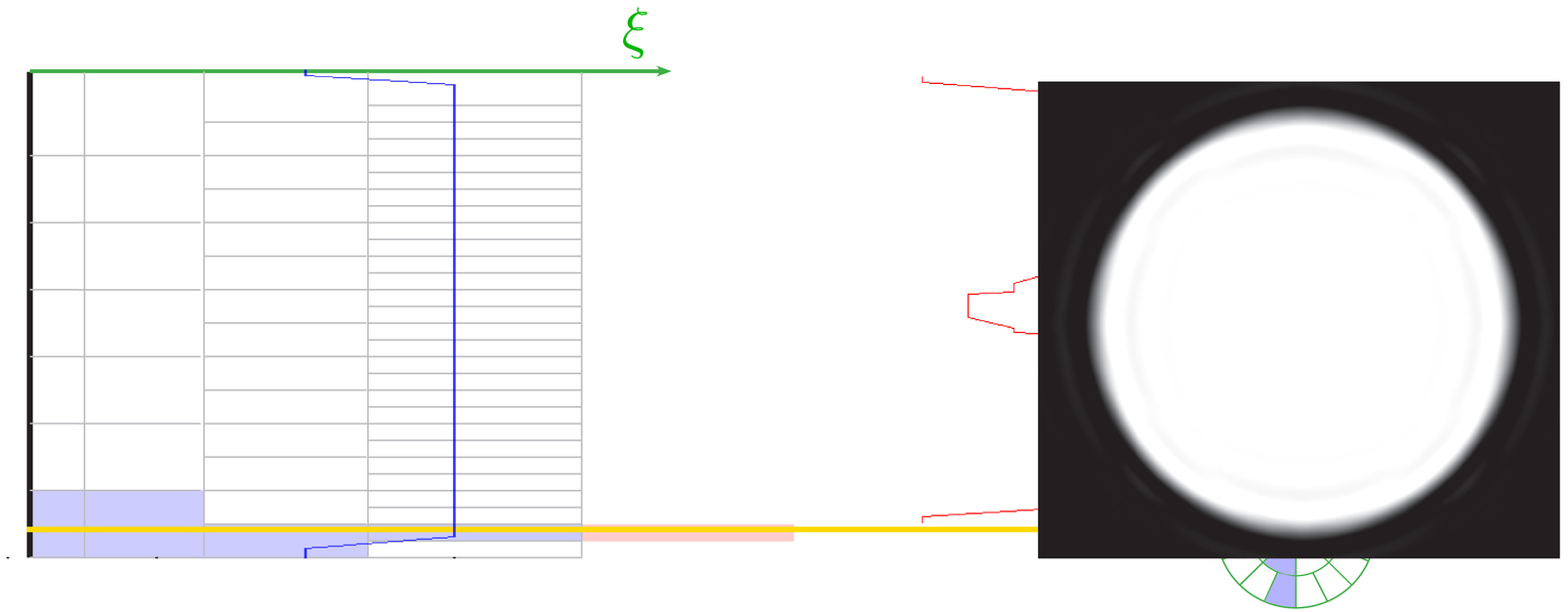


Image generation

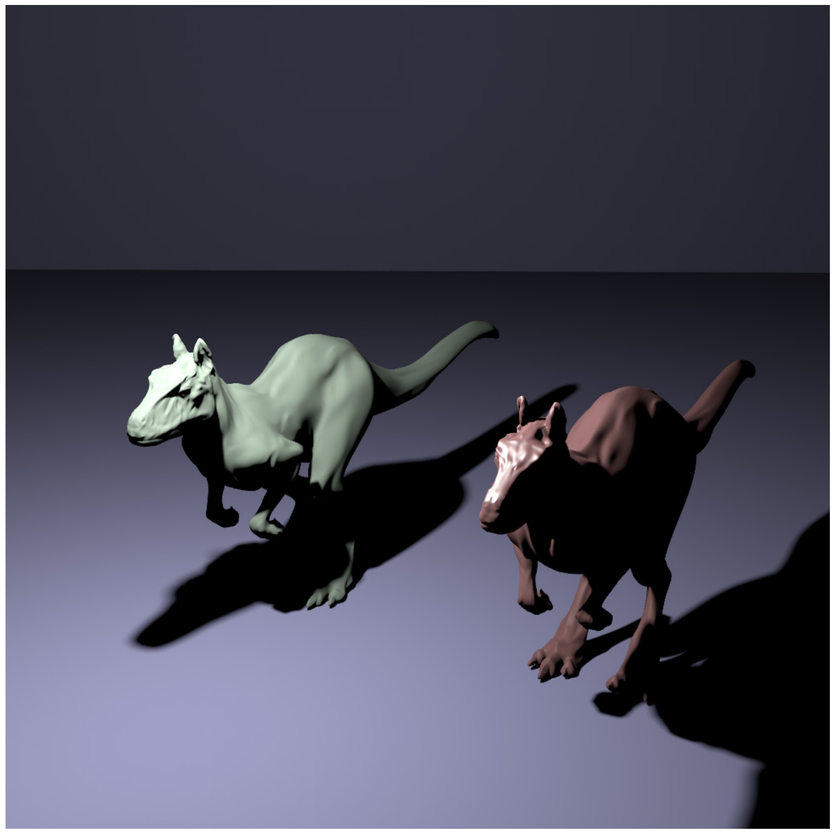
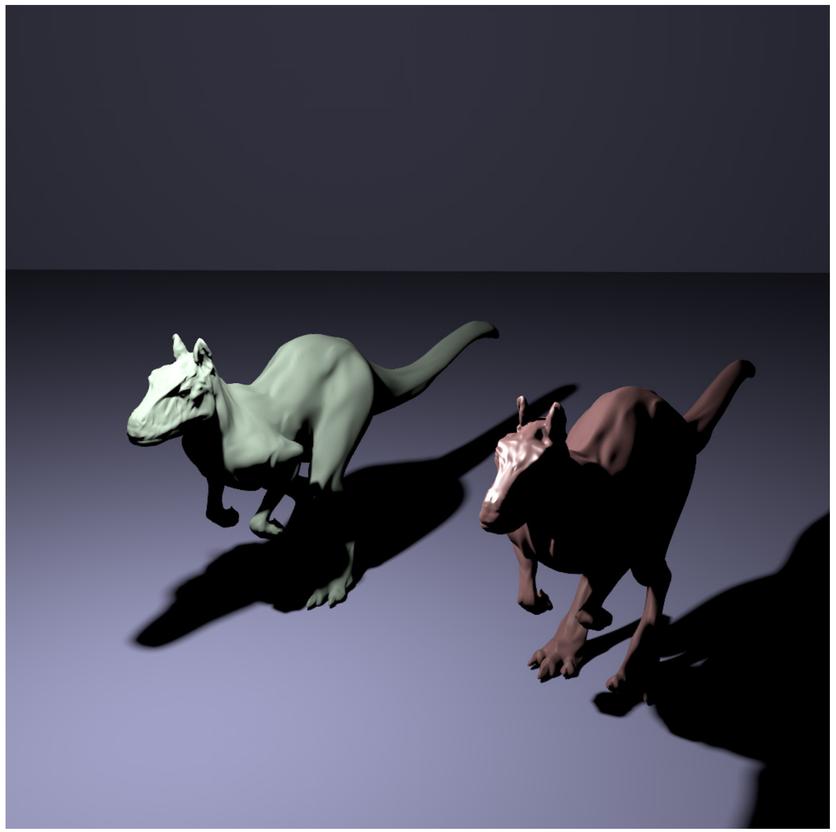


Image generation



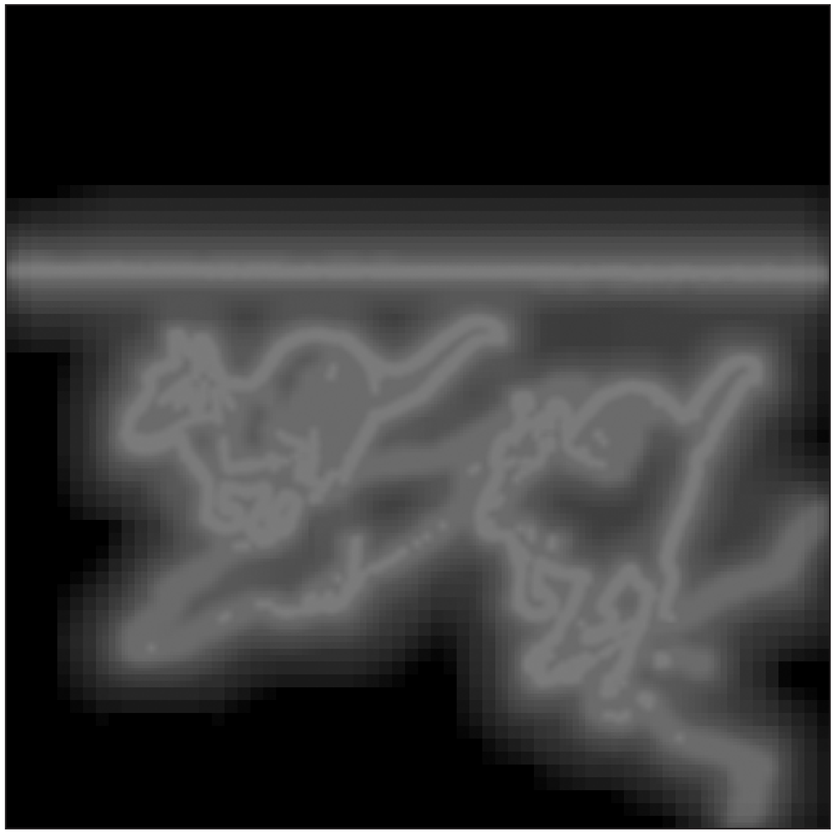
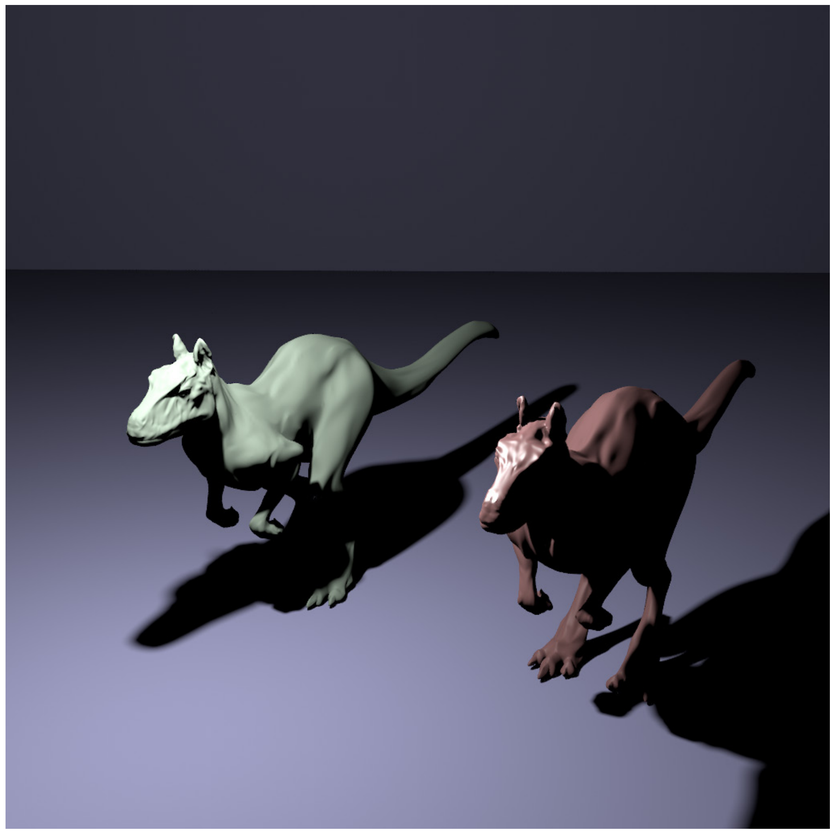


Image generation



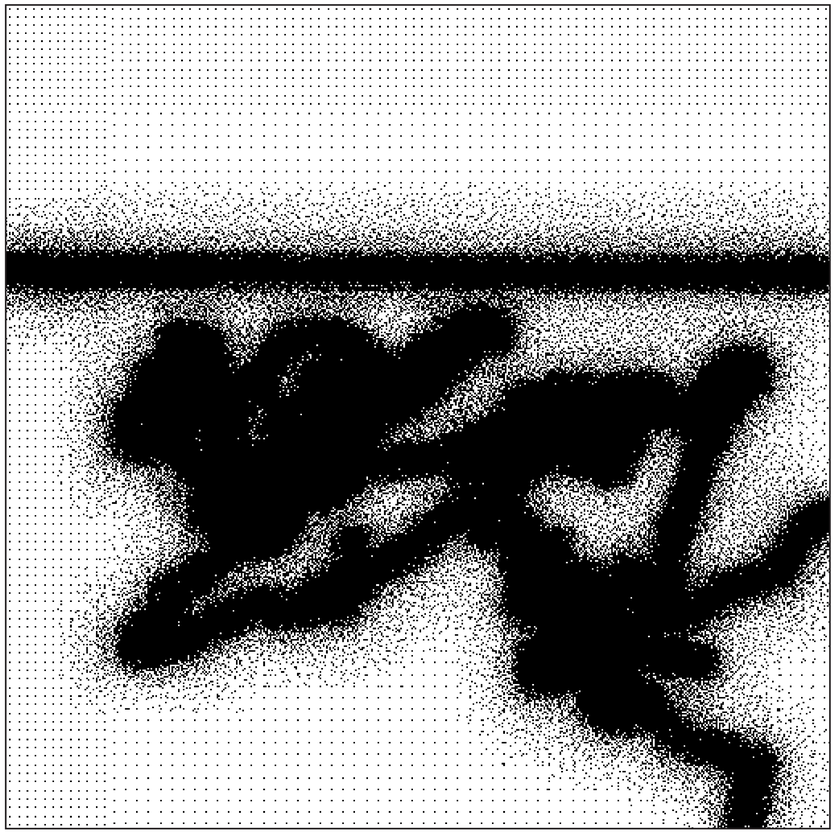
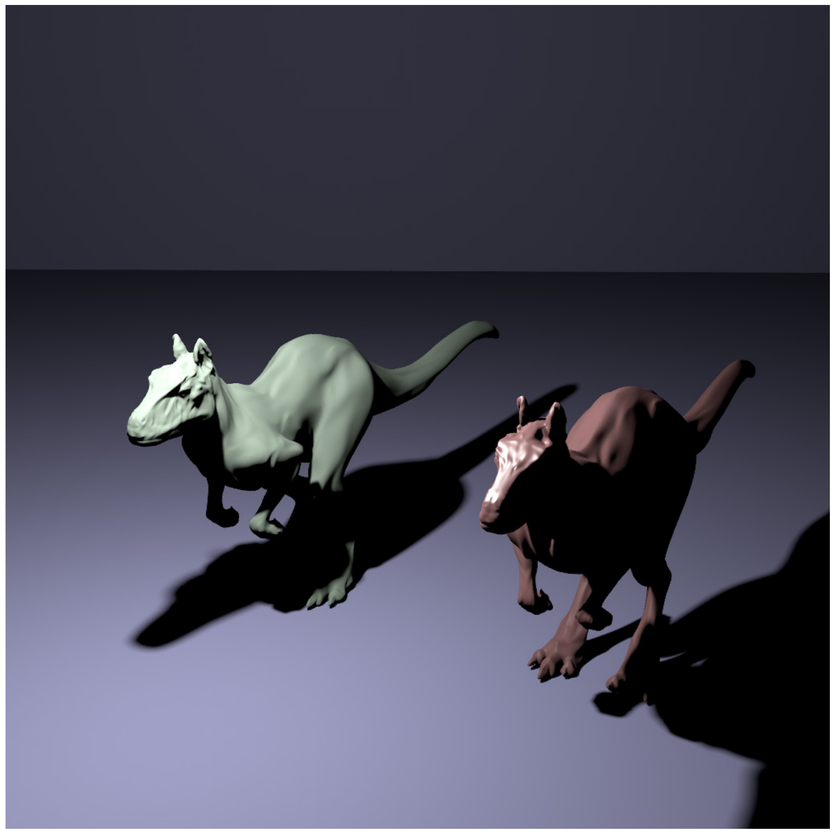


Image generation



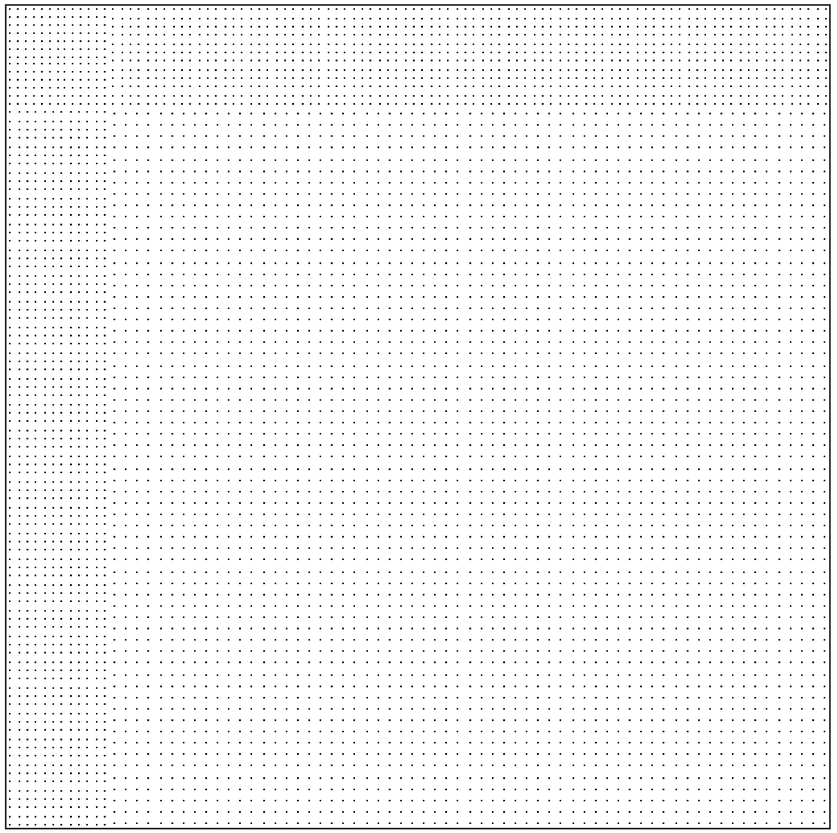
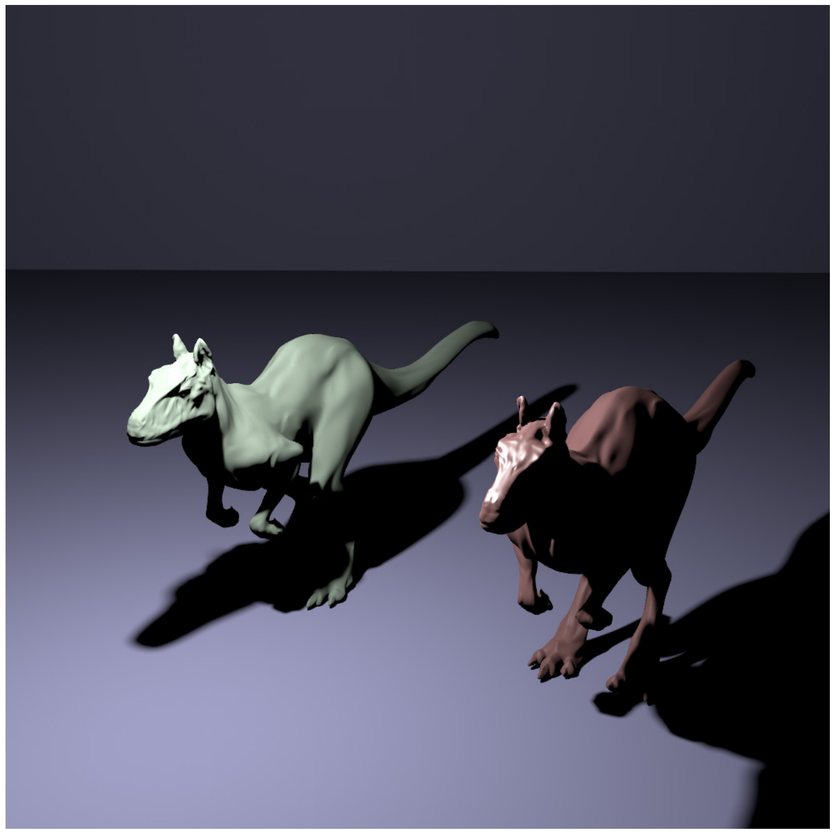


Image generation



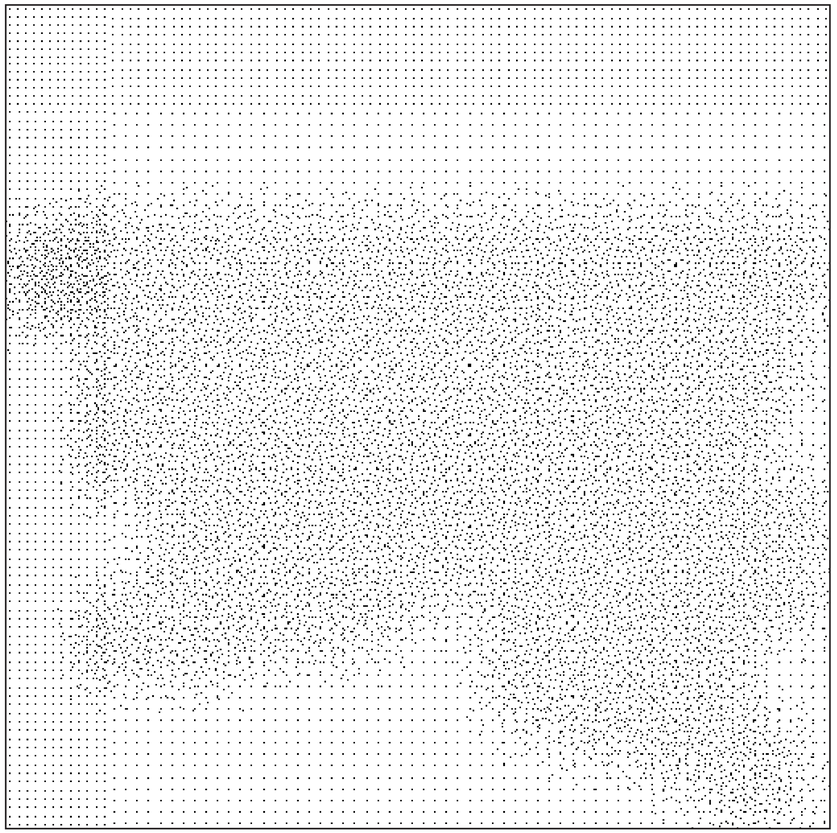
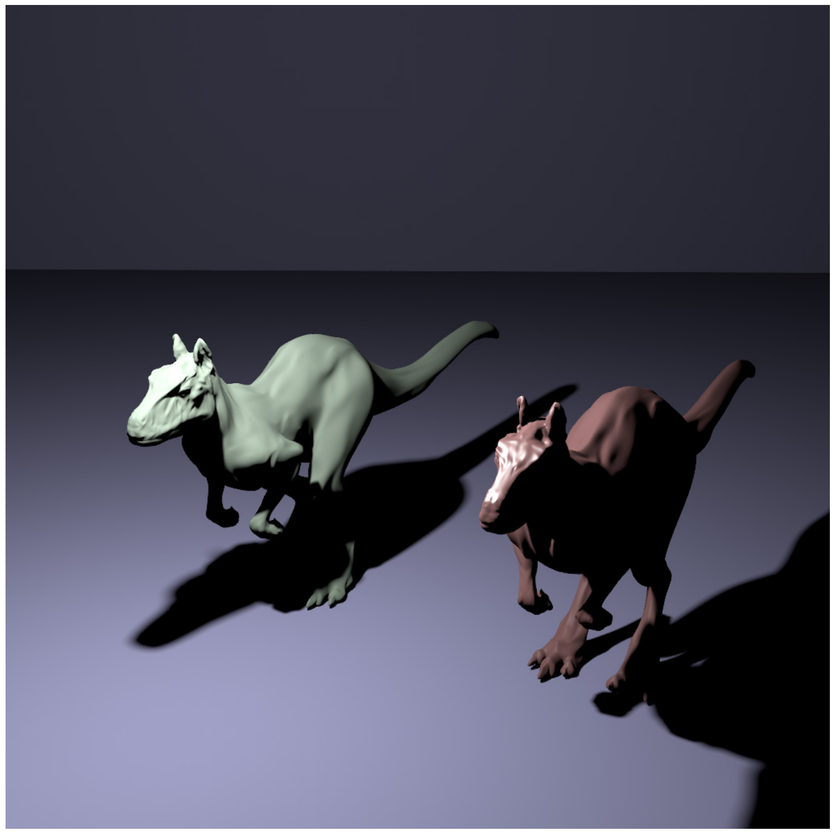


Image generation



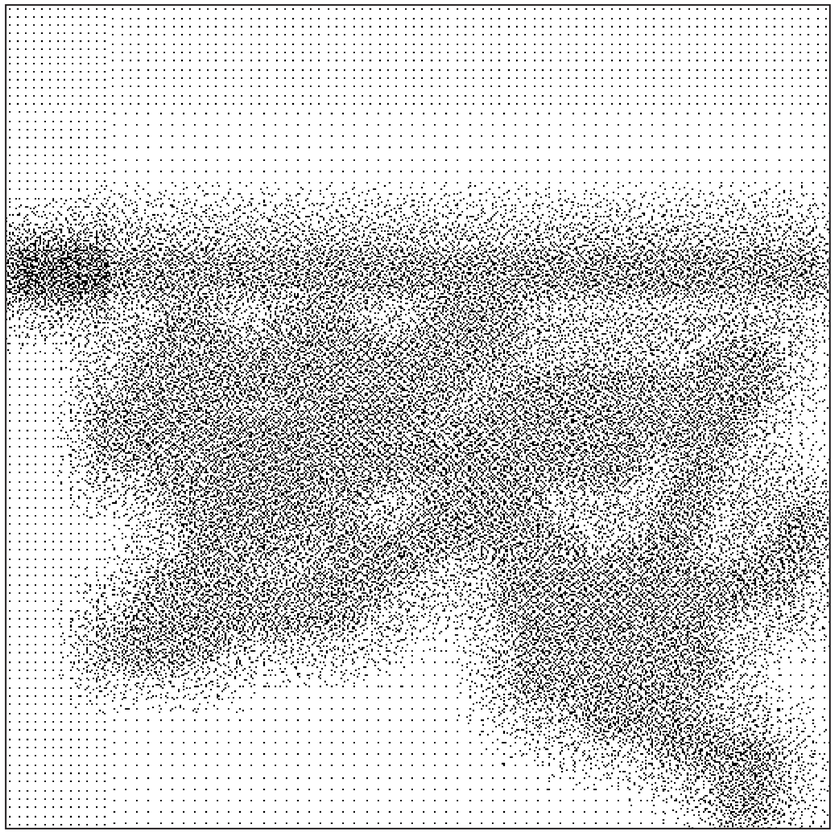
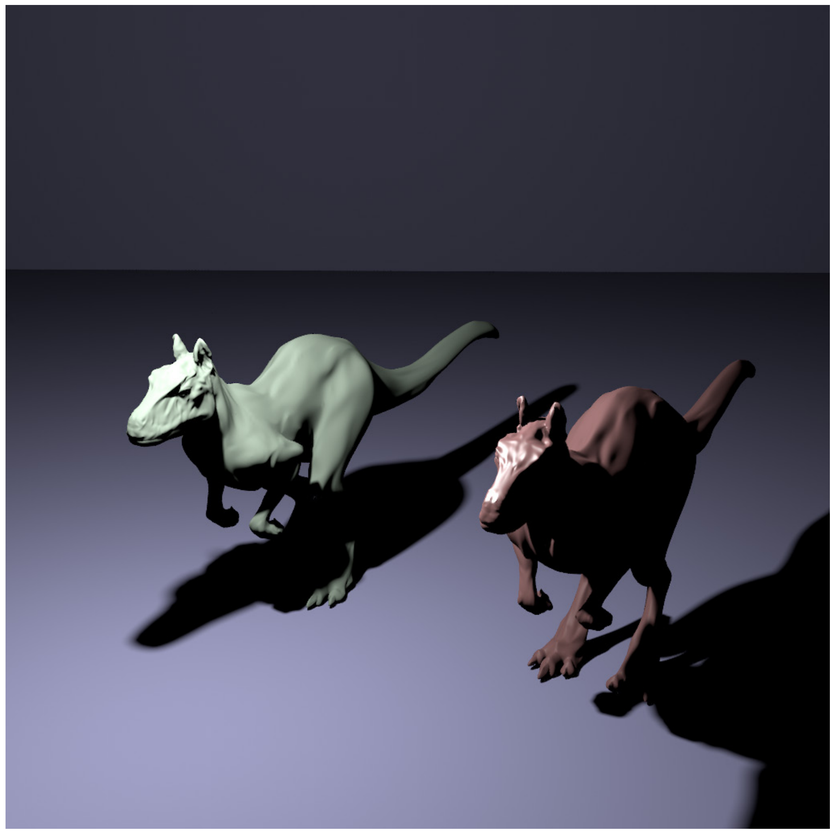


Image generation



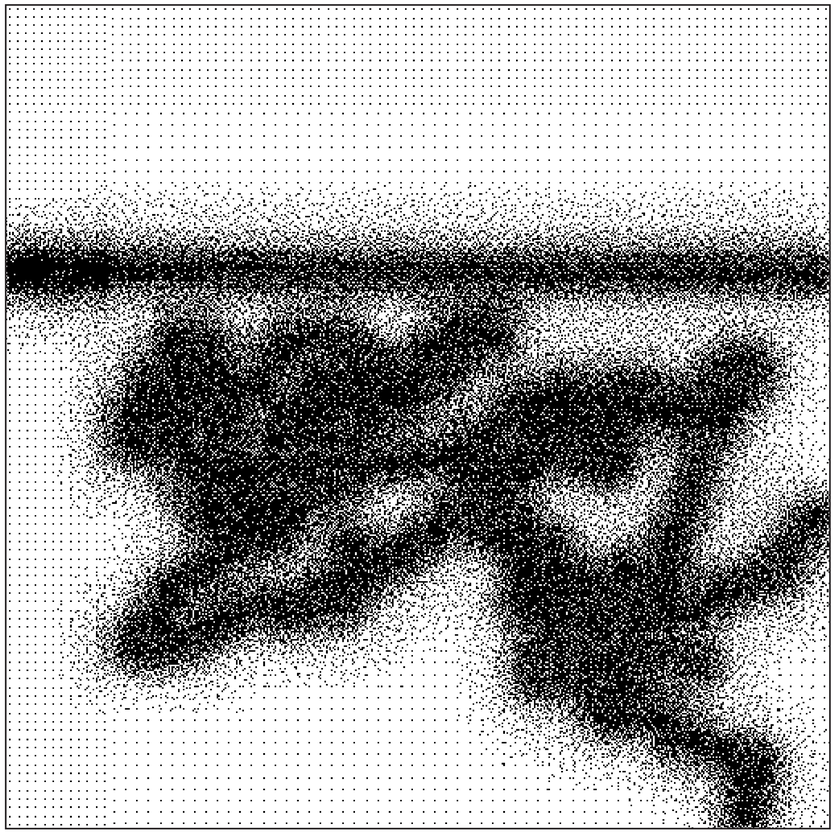
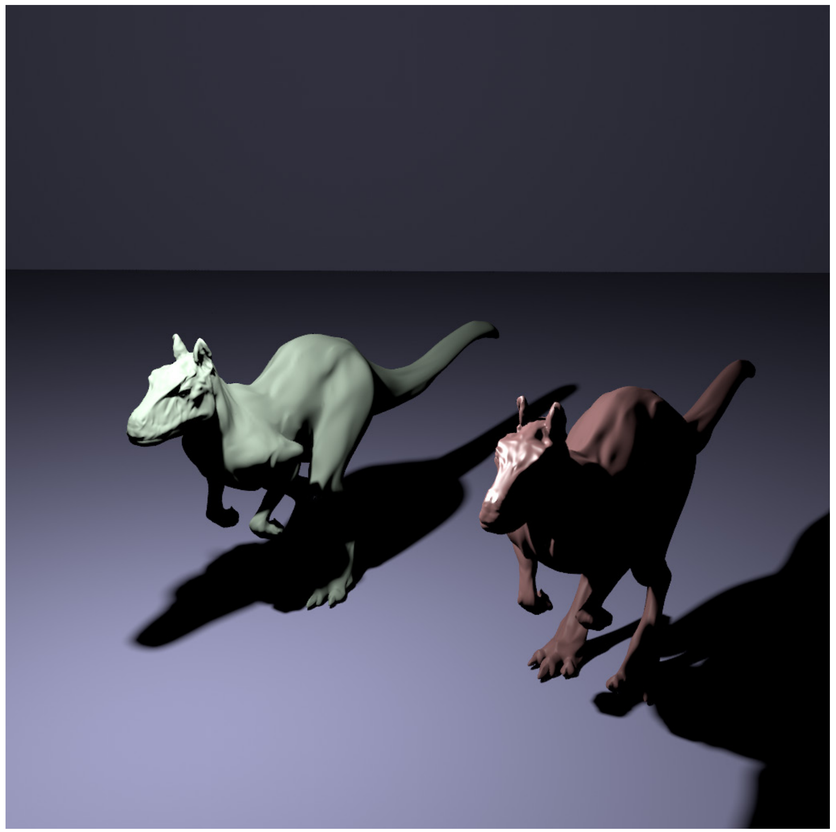


Image generation



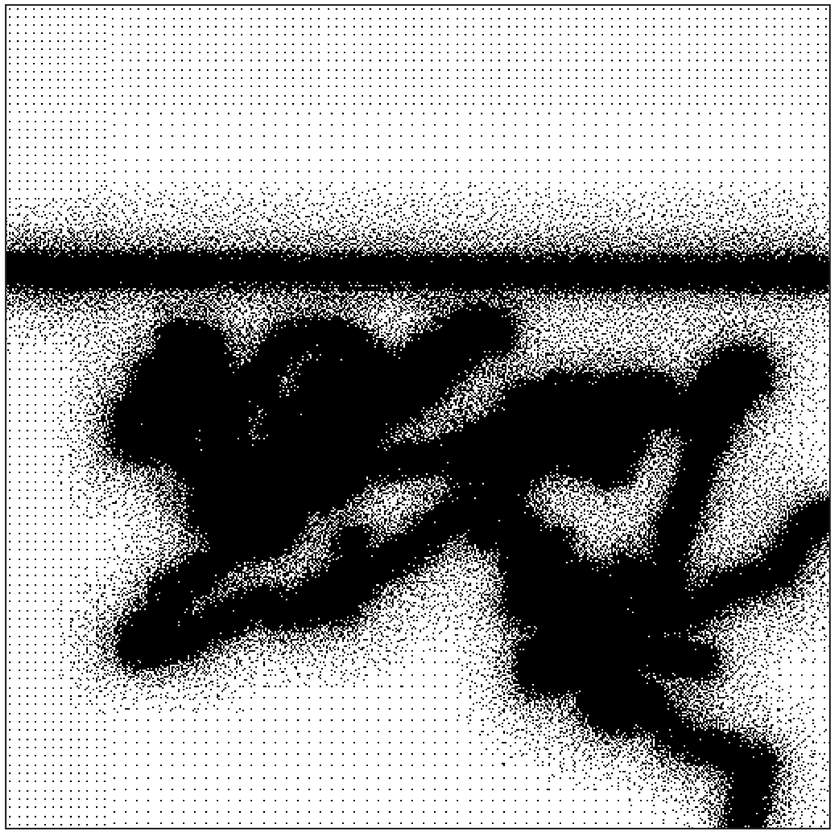
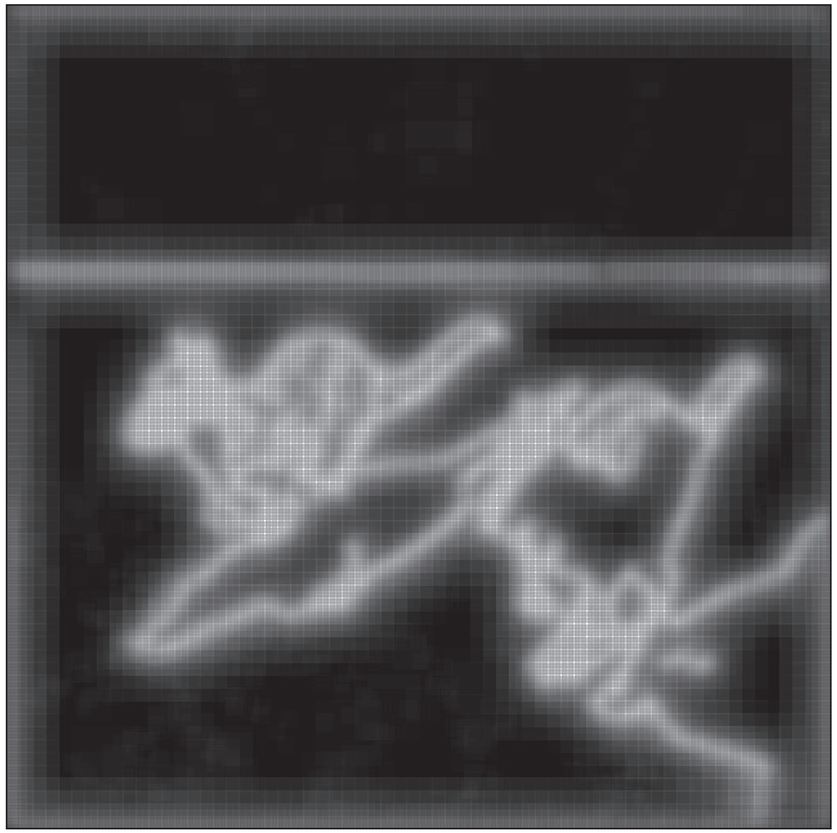


Image generation



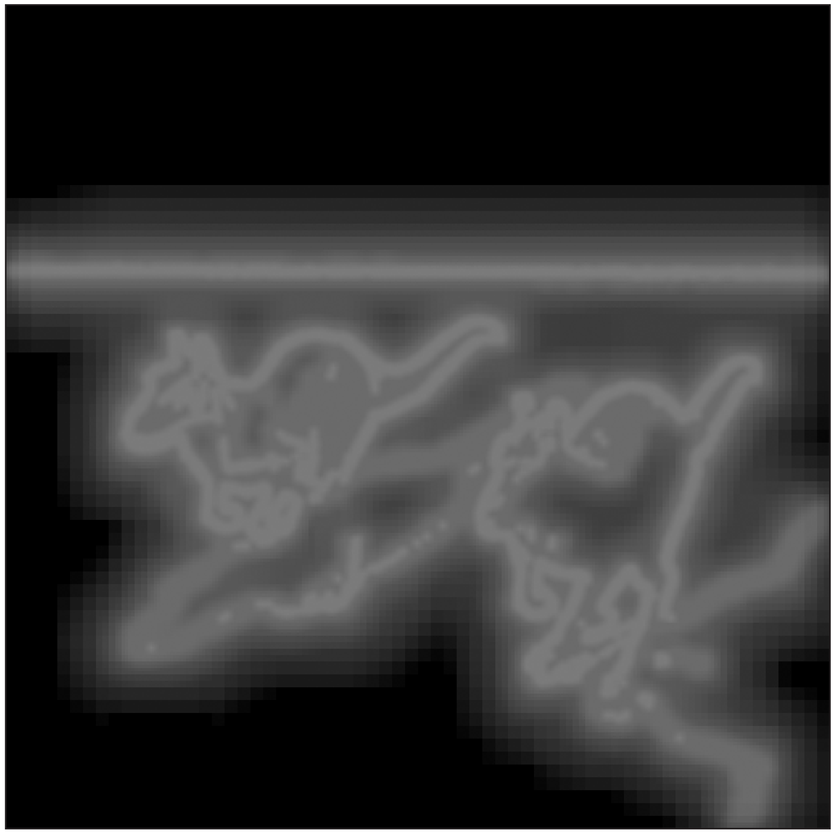
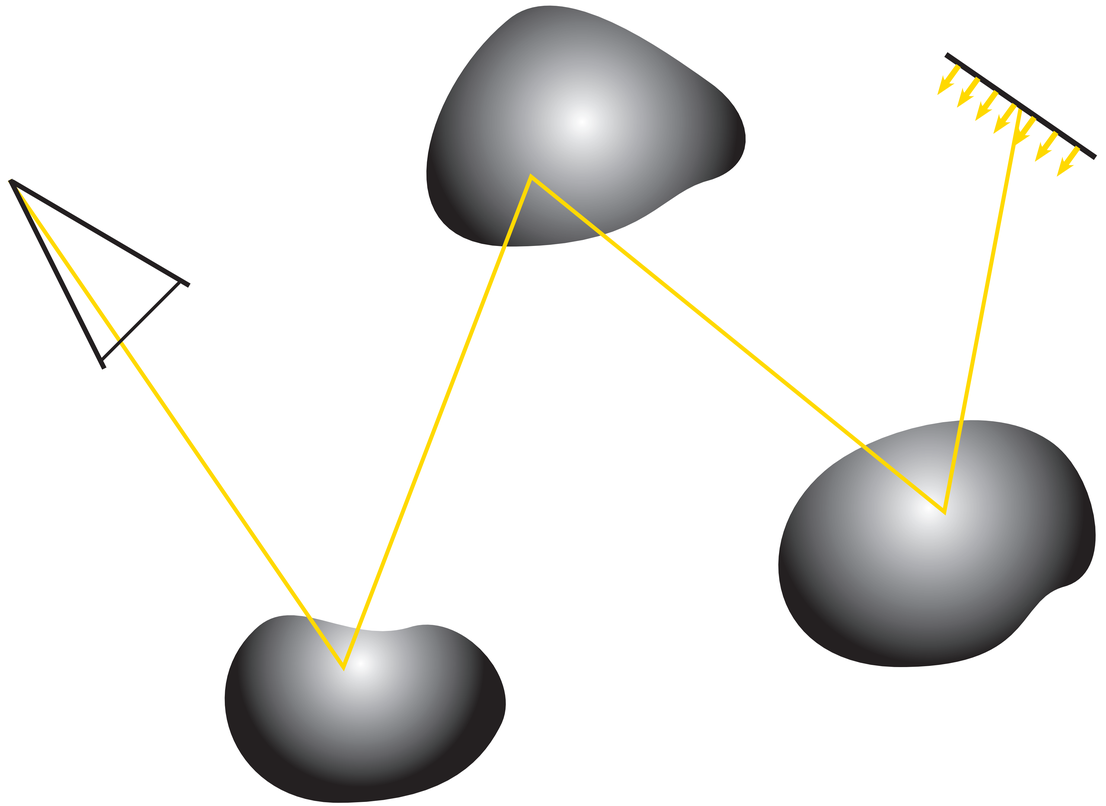


Image generation

1. Find (approximate) tight function space
2. Sample function
3. Construct reproducing kernel basis

Image generation



How many rays do we need?

How many rays do we need?

» As many as there are nonzero coefficients in the sparsest signal representation (times a small constant). «

How many rays do we need?

» As many as there are nonzero coefficients in the sparsest signal representation (times a small constant). «

Recipe:

1. Find (approximate) tight function space
2. Sample function
3. Construct reproducing kernel basis

What has this all to do with deep learning?

What has this all to do with deep learning?

arXiv:1608.08225v1 [cond-mat.dis-nn] 29 Aug 2016

Why does deep and cheap learning work so well?

Henry W. Lin and Max Tegmark
Dept. of Physics, Harvard University, Cambridge, MA 02138 and
Dept. of Physics & MIT Kavli Institute, Massachusetts Institute of Technology, Cambridge, MA 02139

(Dated: August 31, 2016)

We show how the success of deep learning depends not only on mathematics but also on physics: although well-known mathematical theorems guarantee that neural networks can approximate arbitrary functions well, the class of functions of practical interest can be approximated through "cheap learning" with exponentially fewer parameters than generic ones, because they have simplifying properties tracing back to the laws of physics. The exceptional simplicity of physics-based functions hinges on properties such as symmetry, locality, compositionality and polynomial log-probability, and we explore how these properties translate into exceptionally simple neural networks approximating both natural phenomena such as images and abstract representations thereof such as drawings. We further argue that when the statistical process generating the data is of a certain hierarchical form prevalent in physics and machine-learning, a deep neural network can be more efficient than a shallow one. We formalize these claims using information theory and discuss the relation to renormalization group procedures. Various "no-flattening theorems" show when these efficient deep networks cannot be accurately approximated by shallow ones without efficiency loss — even for linear networks.

I. INTRODUCTION

Deep learning works remarkably well, and has helped dramatically improve the state-of-the-art in areas ranging from speech recognition, translation and visual object recognition to drug discovery, genomics and automatic game playing [1]. However, it is still not fully understood *why* deep learning works so well. In contrast to GOFAI ("good old-fashioned AI") algorithms that are hand-crafted and fully understood analytically, many algorithms using artificial neural networks are understood only at a heuristic level, where we empirically know that certain training protocols employing large data sets will result in excellent performance. This is reminiscent of the situation with human brains: we know that if we train a child according to a certain curriculum, she will learn certain skills — but we lack a deep understanding of how her brain accomplishes this.

This makes it timely and interesting to develop new analytic insights on deep learning and its successes, which is the goal of the present paper. Such improved understanding is not only interesting in its own right, and for potentially providing new clues about how brains work, but it may also have practical applications. Better understanding the shortcomings of deep learning may suggest ways of improving it, both to make it more capable and to make it more robust [2].

A. The swindle: why does "cheap learning" work?

Throughout this paper, we will adopt a physics perspective on the problem, to prevent application-specific details from obscuring simple general results related to dynamics, symmetries, renormalization, *etc.*, and to exploit useful similarities between deep learning and statistical mechanics.

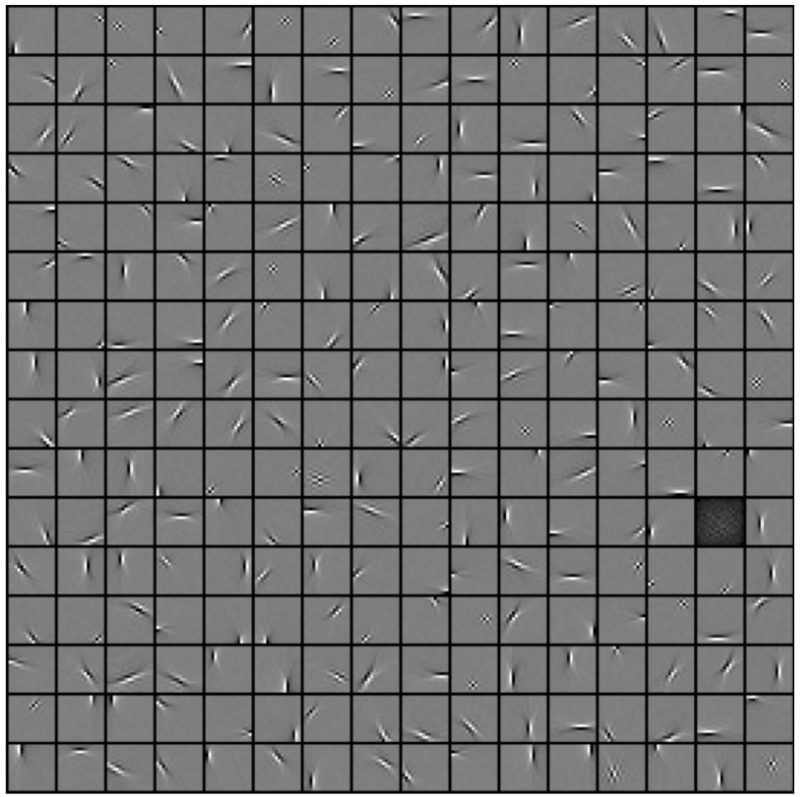
For concreteness, let us focus on the task of approximating functions. As illustrated in Figure 1, this covers most core sub-fields of machine learning, including unsupervised learning, classification and prediction. For example, if we are interested in classifying faces, then we may want our neural network to implement a function where we feed in an image represented by a million greyscale pixels and get as output the probability distribution over a set of people that the image might represent.

FIG. 1: Neural networks can approximate probability distributions. Given many samples of random vectors $\mathbf{x}$ and $\mathbf{y}$, both classification and prediction involve viewing $\mathbf{y}$ as a stochastic function of $\mathbf{x}$ and attempting to estimate the probability distributions for $\mathbf{x}$ given $\mathbf{y}$ and $\mathbf{y}$ given $\mathbf{x}$, respectively. In contrast, unsupervised learning attempts to approximate the joint probability distribution of $\mathbf{x}$ and $\mathbf{y}$ without making any assumptions about causality. In all three cases, the neural network searches for patterns in the data that can be used to better model the probability distribution.

When investigating the quality of a neural net, there are

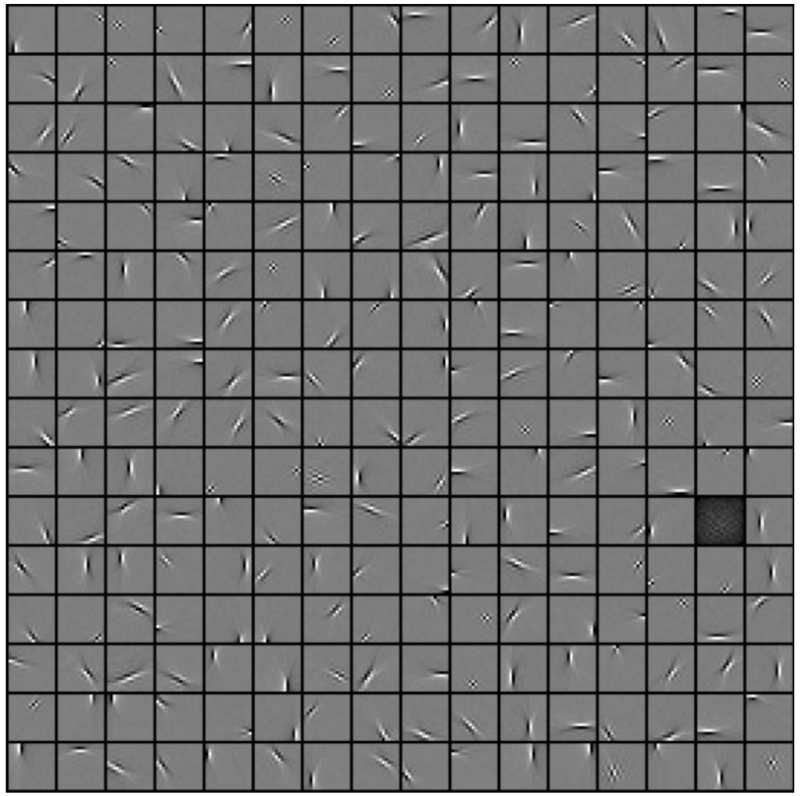
H. W. Lin and M. Tegmark, *Why does deep and cheap learning work so well?*, Aug. 2016.

What has this all to do with deep learning?

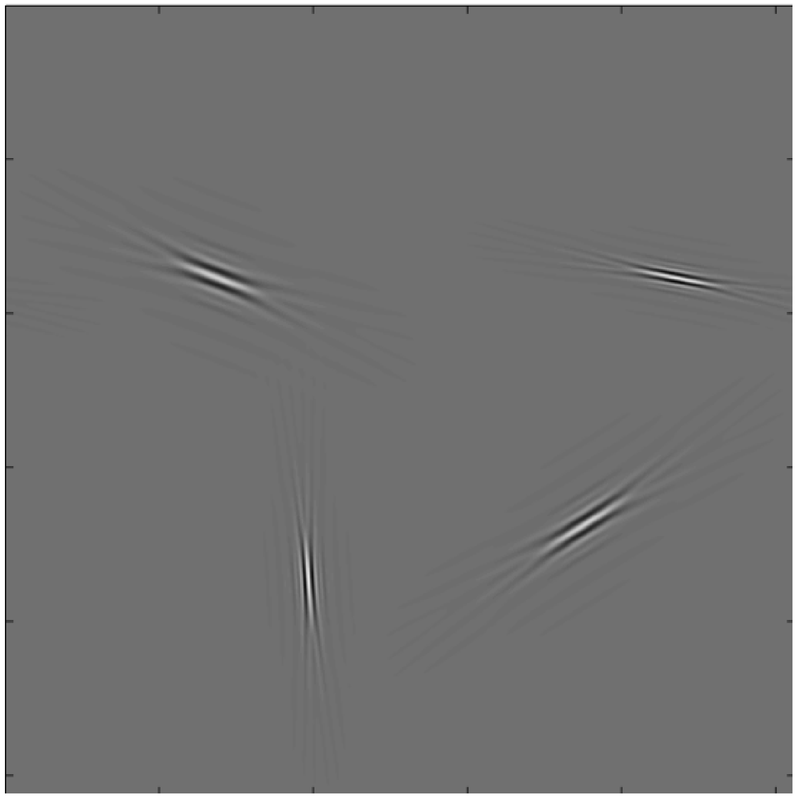


M. A. Ranzato, A. Krizhevsky, and G. E. Hinton, *Factored 3-Way Restricted Boltzmann Machines For Modeling Natural Images*, in AISTATS2010, 2010.

What has this all to do with deep learning?



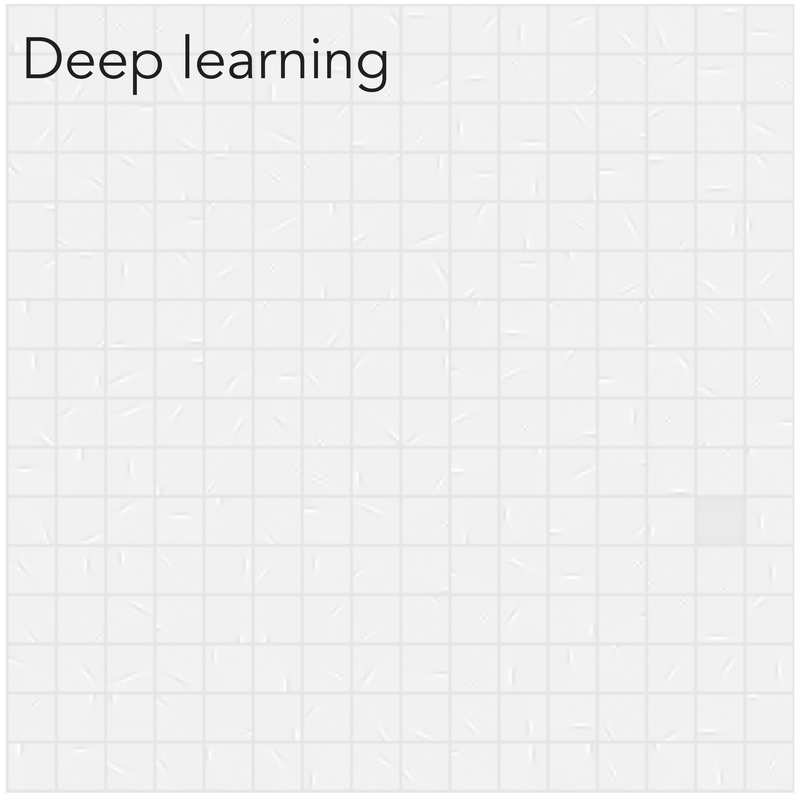
M. A. Ranzato, A. Krizhevsky, and G. E. Hinton, “Factored 3-Way Restricted Boltzmann Machines For Modeling Natural Images,” in AISTATS2010, 2010.



J. Ma and G. Plonka, “The Curvelet Transform,” IEEE Signal Process. Mag., vol. 27, no. 2, pp. 118–133, Mar. 2010.

What has this all to do with deep learning?

Deep learning



from (M. A. Ranzato, A. Krizhevsky, and G. E. Hinton, "Factored 3-Way Restricted Boltzmann Machines For Modeling Natural Images," in AISTATS2010, 2010.

Local frequency analysis

J. Ma and G. Plonka, "The Curvelet Transform," IEEE Signal Process. Mag., vol. 27, no. 2, pp. 118–133, Mar. 2010.

What has this all to do with deep learning?

Deep learning

Explanation: Renormalization group

E.g. P. Mehta and D. J. Schwab, "An exact mapping between the Variational Renormalization Group and Deep Learning," Oct. 2014.

from (M. A. Ranzato, A. Krizhevsky, and G. E. Hinton, "Factored 3-Way Restricted Boltzmann Machines For Modeling Natural Images," in AISTATS2010, 2010.

Local frequency analysis

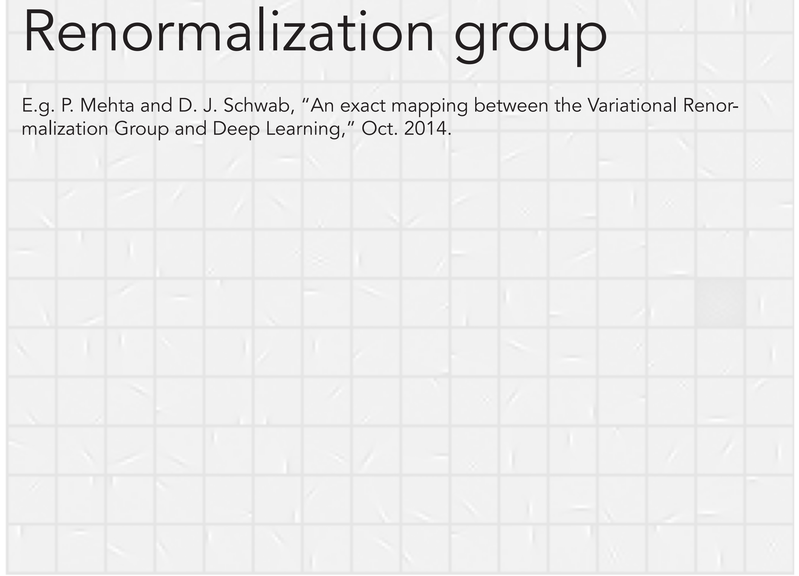
J. Ma and G. Plonka, "The Curvelet Transform," IEEE Signal Process. Mag., vol. 27, no. 2, pp. 118–133, Mar. 2010.

What has this all to do with deep learning?

Deep learning

Explanation:
Renormalization group

E.g. P. Mehta and D. J. Schwab, “An exact mapping between the Variational Renormalization Group and Deep Learning,” Oct. 2014.

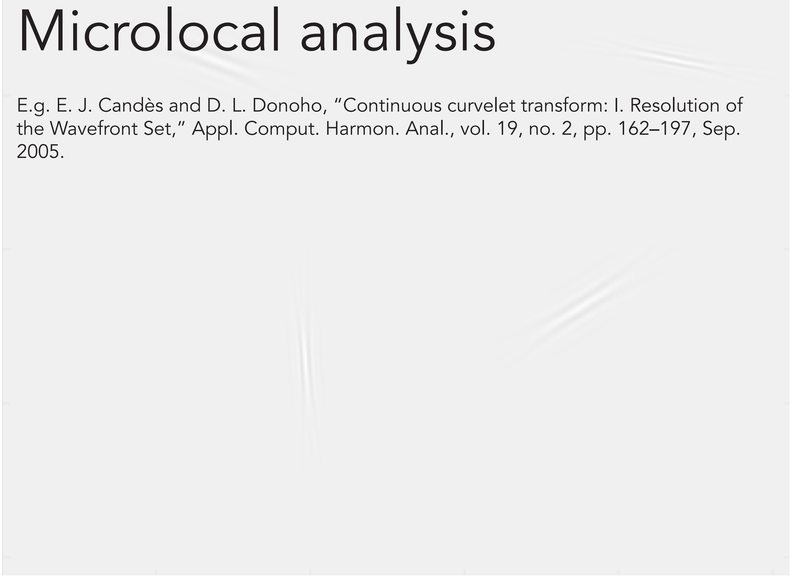


from (M. A. Ranzato, A. Krizhevsky, and G. E. Hinton, “Factored 3-Way Restricted Boltzmann Machines For Modeling Natural Images,” in AISTATS2010, 2010.

Local frequency analysis

Explanation:
Microlocal analysis

E.g. E. J. Candès and D. L. Donoho, “Continuous curvelet transform: I. Resolution of the Wavefront Set,” Appl. Comput. Harmon. Anal., vol. 19, no. 2, pp. 162–197, Sep. 2005.



J. Ma and G. Plonka, “The Curvelet Transform,” IEEE Signal Process. Mag., vol. 27, no. 2, pp. 118–133, Mar. 2010.

What has this all to do with deep learning?

Deep learning

Explanation:
Renormalization group

E.g. P. Mehta and D. J. Schwab, “An exact mapping between the Variational Renormalization Group and Deep Learning,” Oct. 2014.

from (M. A. Ranzato, A. Krizhevsky, and G. E. Hinton, “Factored 3-Way Restricted Boltzmann Machines For Modeling Natural Images,” in AISTATS2010, 2010.

Local frequency analysis

Explanation:
Microlocal analysis

E.g. E. J. Candès and D. L. Donoho, “Continuous curvelet transform: I. Resolution of the Wavefront Set,” Appl. Comput. Harmon. Anal., vol. 19, no. 2, pp. 162–197, Sep. 2005.

J. Ma and G. Plonka, “The Curvelet Transform,” IEEE Signal Process. Mag., vol. 27, no. 2, pp. 118–133, Mar. 2010.

Quasi eigenfunctions for light transport

What has this all to do with deep learning?

Deep learning

Explanation:
Renormalization group

E.g. P. Mehta and D. J. Schwab, "An exact mapping between the Variational Renormalization Group and Deep Learning," Oct. 2014.

Connection?

Local frequency analysis

Explanation:
Microlocal analysis

E.g. E. J. Candès and D. L. Donoho, "Continuous curvelet transform: I. Resolution of the Wavefront Set," Appl. Comput. Harmon. Anal., vol. 19, no. 2, pp. 162–197, Sep. 2005.

from (M. A. Ranzato, A. Krizhevsky, and G. E. Hinton, "Factored 3-Way Restricted Boltzmann Machines For Modeling Natural Images," in AISTATS2010, 2010.

J. Ma and G. Plonka, "The Curvelet Transform," IEEE Signal Process. Mag., vol. 27, no. 2, pp. 118–133, Mar. 2010.

Quasi eigenfunctions for light transport

What has this all to do with deep learning?

Deep learning

Explanation: Renormalization group

E.g. P. Mehta and D. J. Schwab, "An exact mapping between the Variational Renormalization Group and Deep Learning," Oct. 2014.

Local frequency analysis

Explanation: Microlocal analysis

E.g. E. J. Candès and D. L. Donoho, "Continuous curvelet transform: I. Resolution of the Wavefront Set," Appl. Comput. Harmon. Anal., vol. 19, no. 2, pp. 162–197, Sep. 2005.

Fredenhagen and co-workers: the renormalization groups arise from microlocal analysis

E.g. R. Brunetti, M. Dütsch, and K. Fredenhagen, "Perturbative algebraic quantum field theory and the renormalization groups," Adv. Theor. Math. Phys., vol. 13, no. 5, pp. 1541–1599, 2009.

from (M. A. Ranzato, A. Krizhevsky, and G. E. Hinton, "Factored 3-Way Restricted Boltzmann Machines For Modeling Natural Images," in AISTATS2010, 2010.

J. Ma and G. Plonka, "The Curvelet Transform," IEEE Signal Process. Mag., vol. 27, no. 2, pp. 118–133, Mar. 2010.

Quasi eigenfunctions for light transport

How many rays do we need?

» As many as there are nonzero coefficients in the sparsest signal representation (times a small constant). «

Recipe:

1. Find (approximate) tight function space
2. Sample function
3. Construct reproducing kernel basis

Reproducing kernel Hilbert spaces

$$\int_X f(x)\,dx$$

Reproducing kernel Hilbert spaces

$$\int_X f(x)\,dx = \int_X \sum_{i=1}^{N} f(x_i)\tilde{k}_i(x)\,dx$$

Reproducing kernel Hilbert spaces

$$\int_X f(x)\,dx = \int_X \sum_{i=1}^{N} f(x_i)\tilde{k}_i(x)\,dx$$

$$= \sum_{i=1}^{N} f(x_i) \int_X \tilde{k}_i(x)\,dx$$

Reproducing kernel Hilbert spaces

$$\int_X f(x)\,dx = \int_X \sum_{i=1}^{N} f(x_i)\tilde{k}_i(x)\,dx$$

$$= \sum_{i=1}^{N} f(x_i) \underbrace{\int_X \tilde{k}_i(x)\,dx}_{w_i}$$

Reproducing kernel Hilbert spaces

$$\int_X f(x)\,dx = \int_X \sum_{i=1}^{N} f(x_i)\tilde{k}_i(x)\,dx$$

$$= \sum_{i=1}^{N} f(x_i) \underbrace{\int_X \tilde{k}_i(x)\,dx}_{w_i}$$

$$= \sum_{i=1}^{N} w_i\, f(x_i)$$