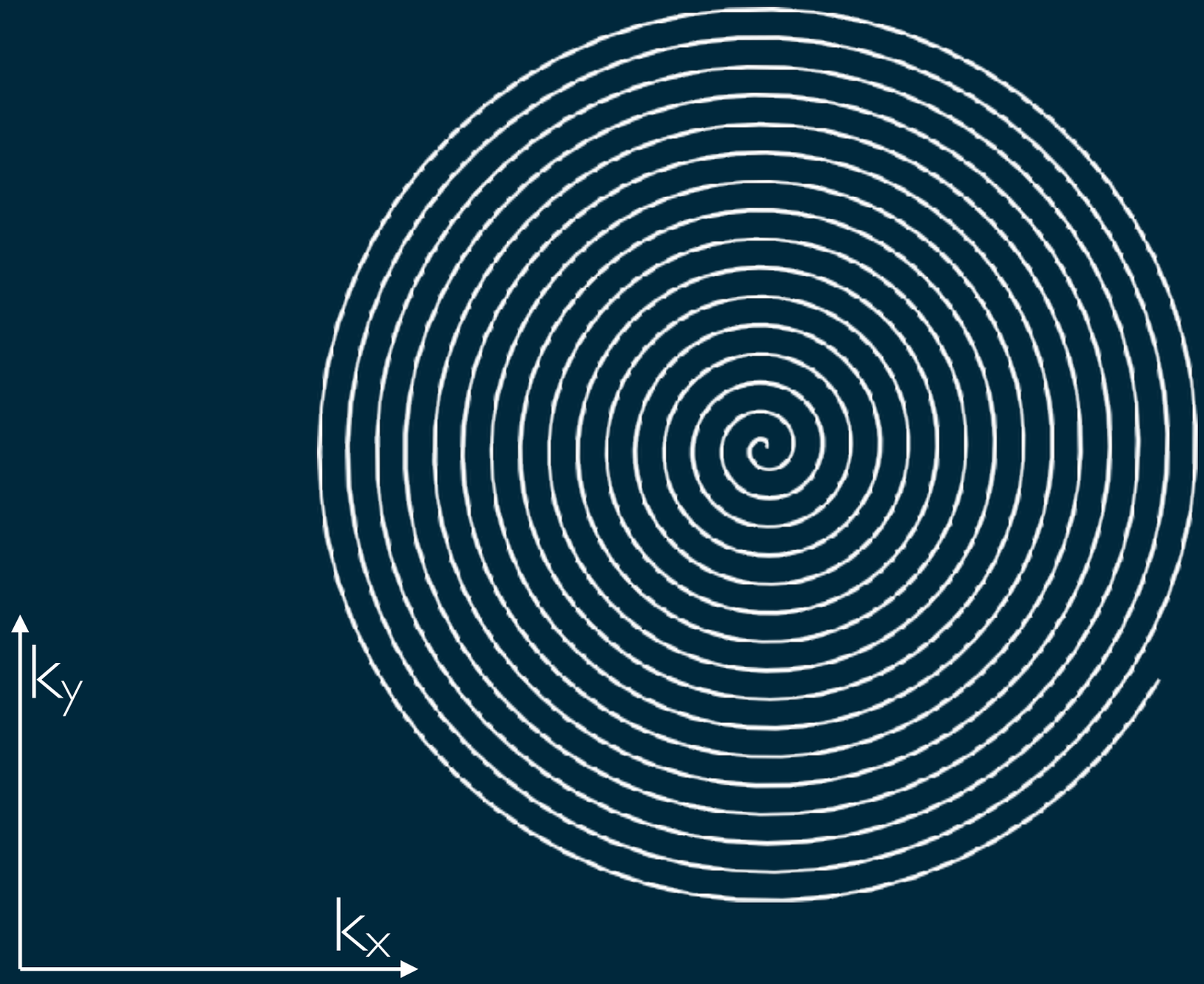
Image Formation Revisited

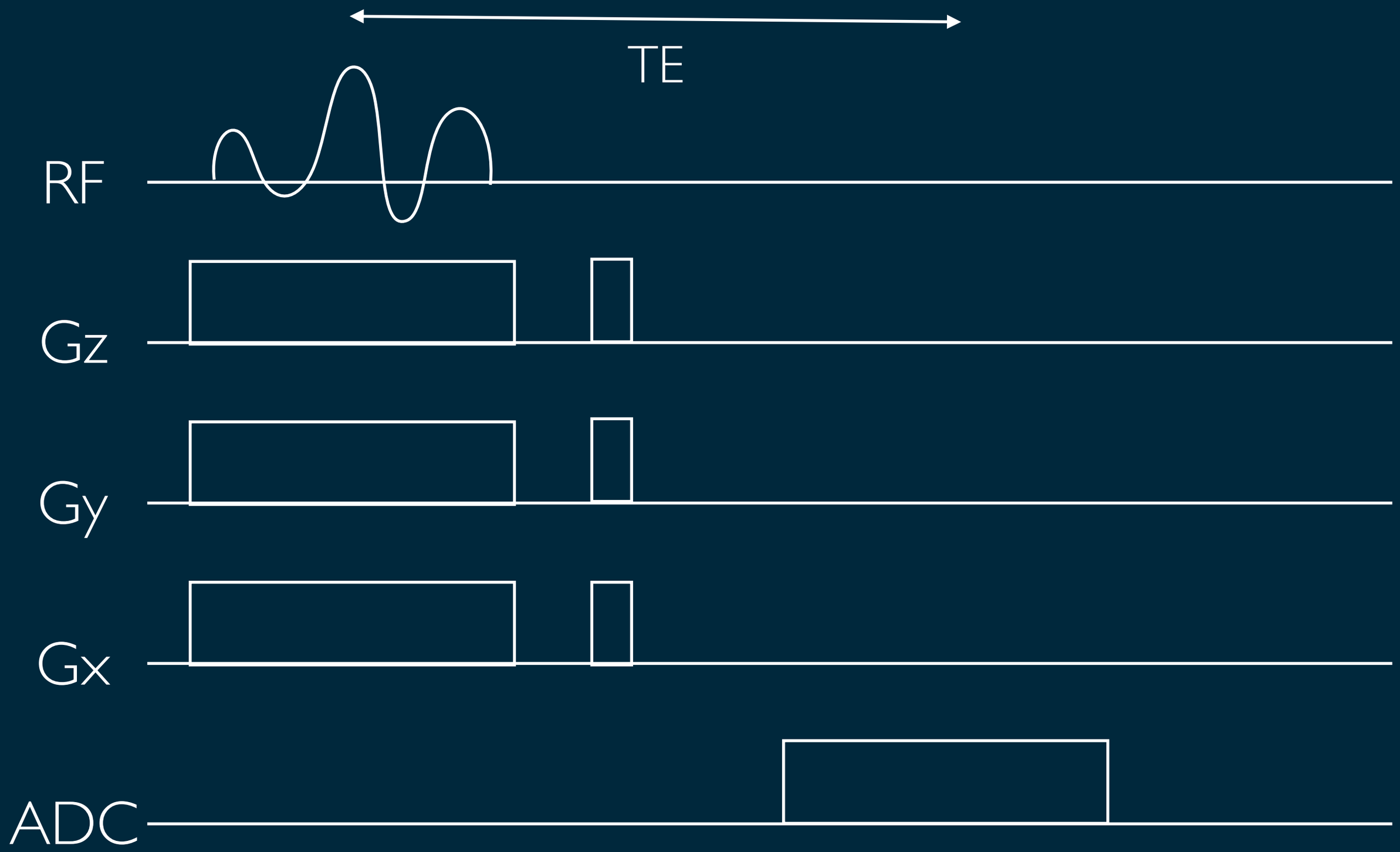
Mark Chiew
mark.chiew@ndcn.ox.ac.uk
Feb. 1, 2018

All Encoding is Phase Encoding

Where is the "frequency encoding" axis?

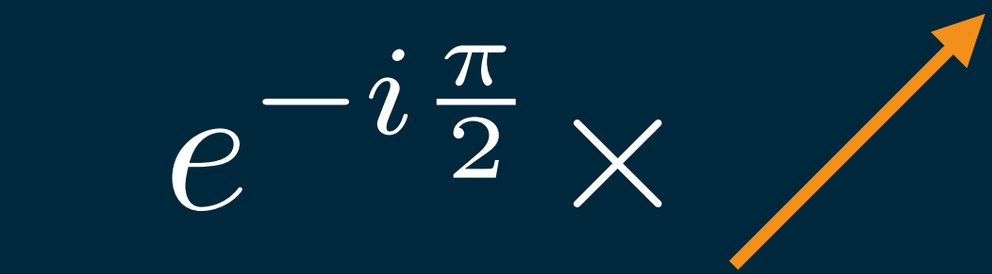


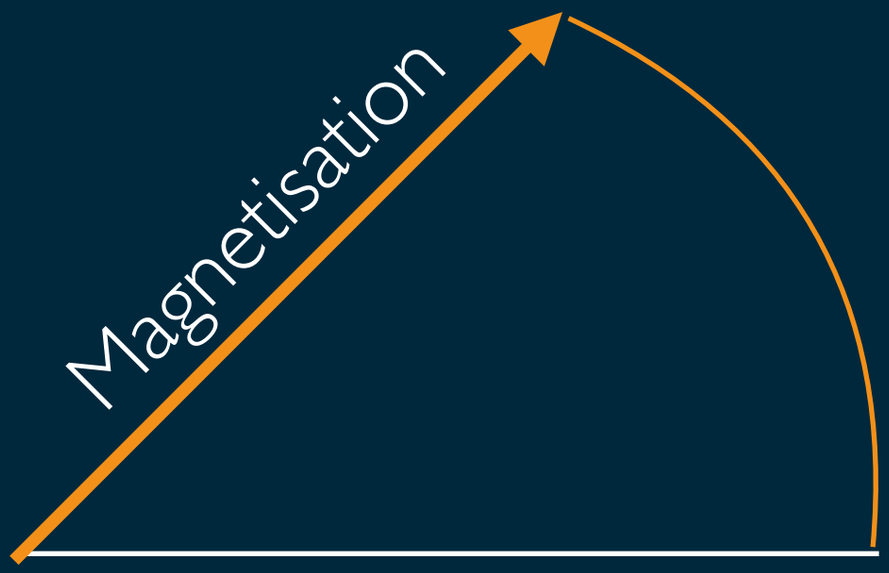
Does this pulse sequence make any sense?



Complex exponentials rotate vectors

$$e^{-i\theta} \times \vec{z} = rotate(\vec{z}, \theta)$$

$$e^{-i\frac{\pi}{2}} \times$$  $$=$$ 



$$\theta = \begin{cases} phase \\ angle \\ rotation \\ orientation \end{cases}$$ in the transverse plane

Frequency is just the phase rate of change

Temporal Frequency

$$\omega = \frac{\partial \theta(x,t)}{\partial t}$$

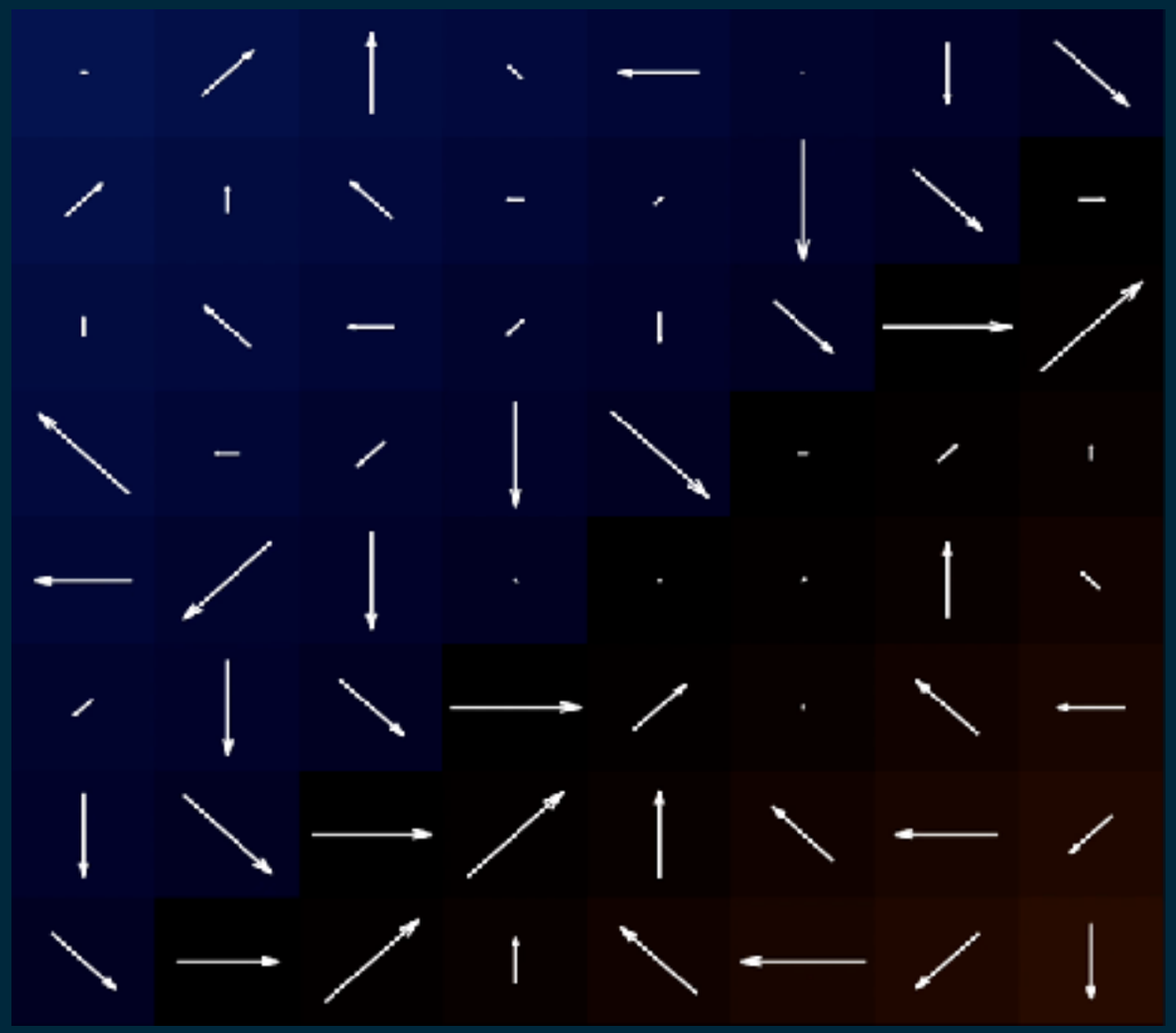
Spatial Frequency

$$k = \frac{\partial \theta(x,t)}{\partial x}$$

How quickly is the phase changing

Magnetisation has magnitude and phase

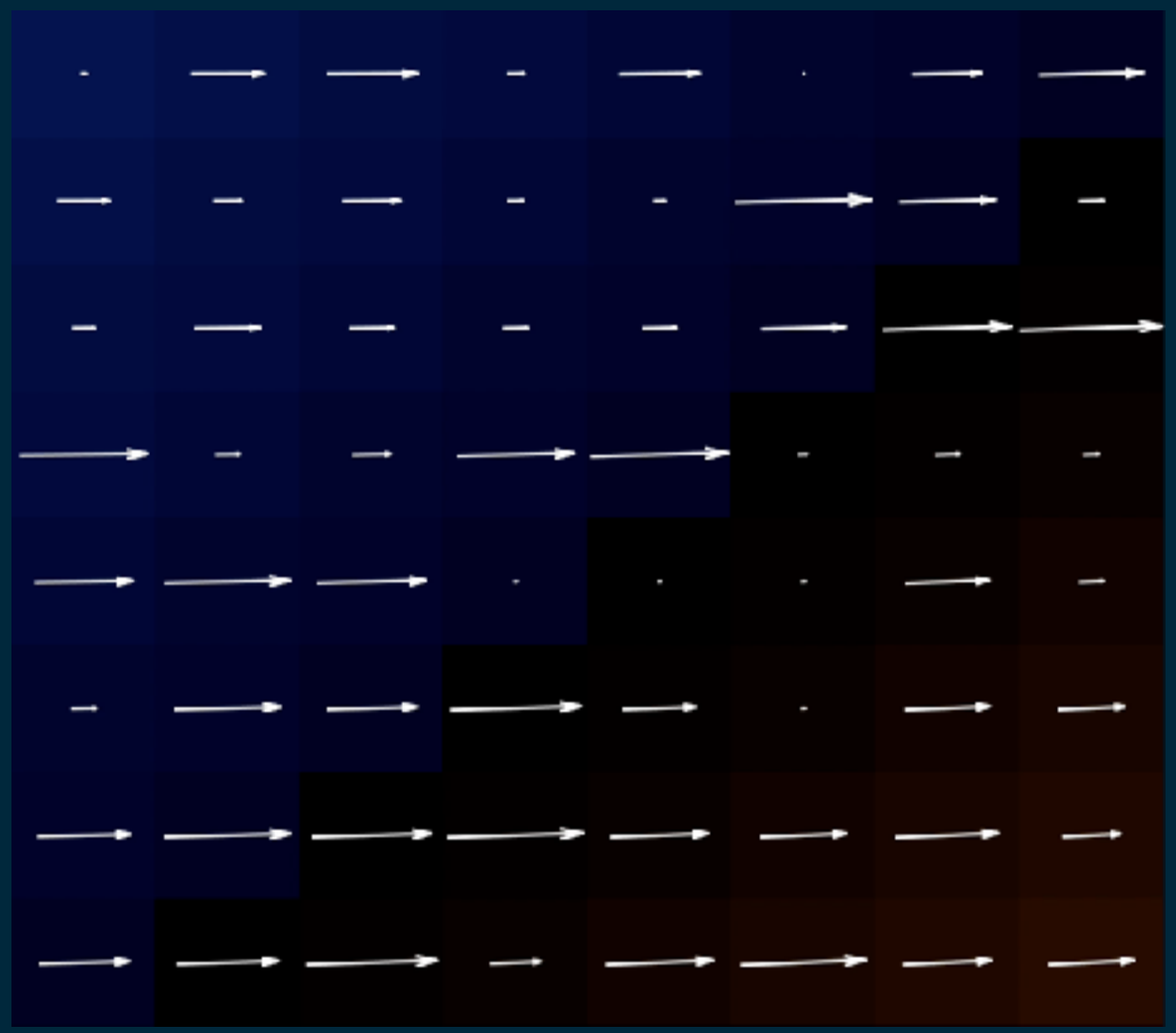
$$M(x,t) = \rho(x)e^{-i\theta(x,t)} \quad [0]$$



Magnetisation Precesses

Magnetisation has magnitude and phase

$$M(x,t) = \rho(x)e^{-i\omega(x)t} \quad [1]$$



Magnetisation Precesses

Larmor Equation

$$\omega = \gamma B \quad [2]$$

Lower Field

Higher Field

Lower Frequency

Higher Frequency

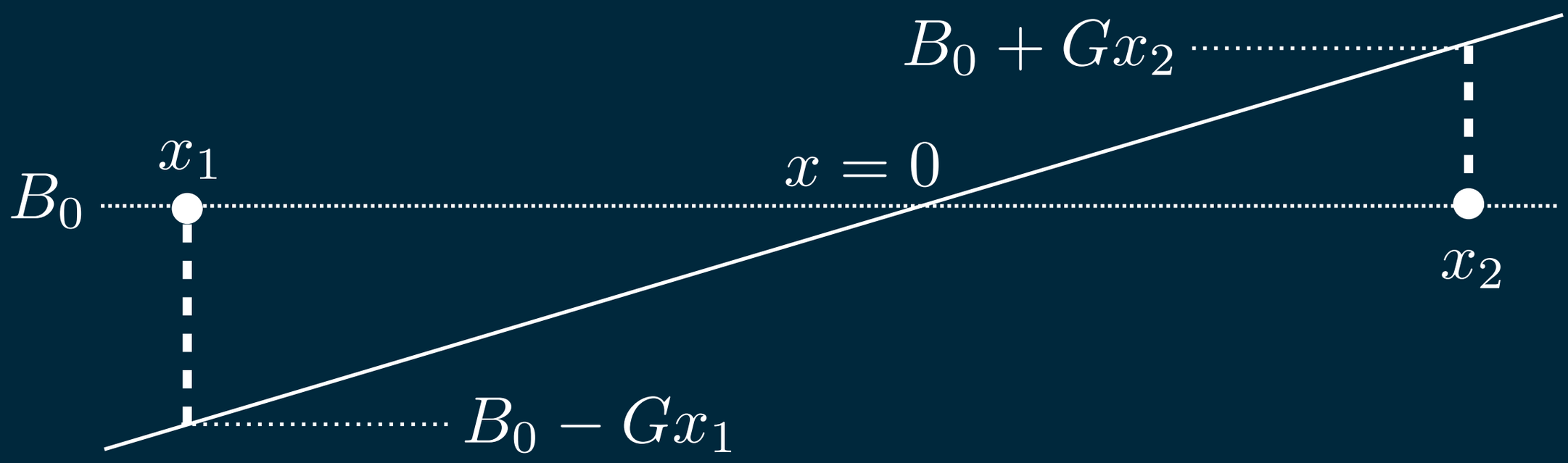
Slower precession/rotation

Faster precession/rotation

Frequency scales with field strength

Linear Magnetic Fields

$$B = B_0 + Gx \quad [3]$$



"Gradients" vary field strength linearly with distance from isocentre

Magnetisation

$$M(x,t) = \rho(x)e^{-i\omega(x)t} \quad [1]$$

Larmor Equation

$$\omega = \gamma B \quad [2]$$

Linear Magnetic Fields

$$B = B_0 + Gx \quad [3]$$

Magnetisation

$$M(x,t) = \rho(x)e^{-i\omega(x)t} \quad [1]$$

Larmor Equation

$$\omega = \gamma B \quad [2]$$

Linear Magnetic Fields

$$B = B_0 + Gx \quad [3]$$

Magnetisation

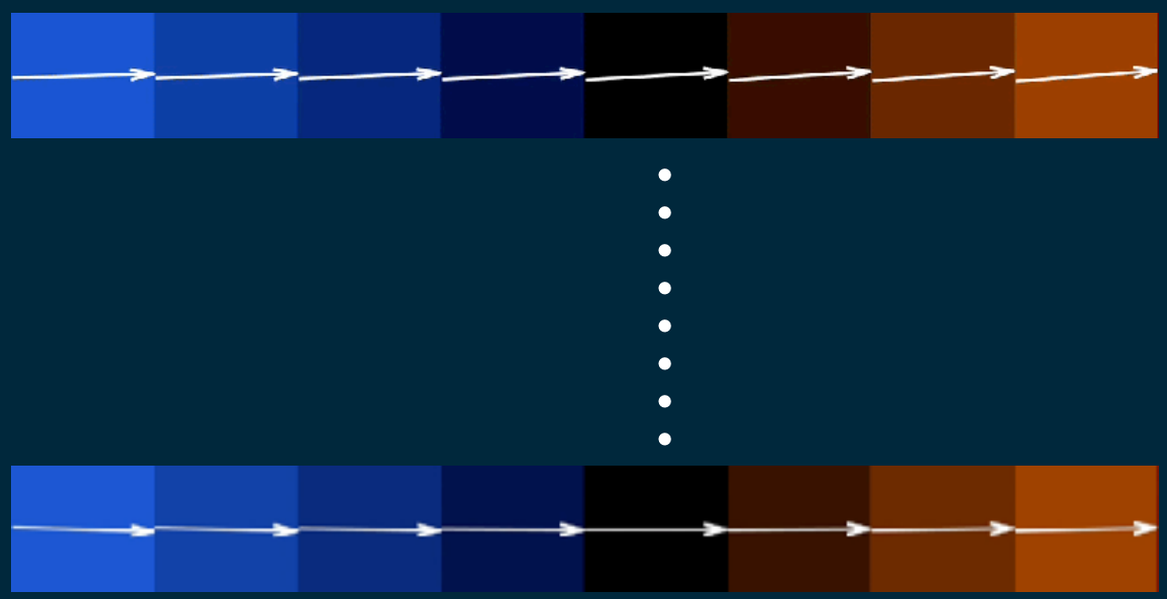
$$M(x,t) = \rho(x)e^{-i\omega(x)t} \quad [1]$$

Frequency varies linearly with position

$$\omega(x) = \gamma B_0 + \gamma(Gx) \quad [4]$$

Magnetisation

$$M(x,t) = \rho(x)e^{-i\gamma B_0 t}e^{-i\gamma G x t} \quad [5]$$



Rotating Frame Magnetisation

$$M(x,t) = M_{rot}(x,t)e^{-i\gamma B_0 t} \quad [6]$$

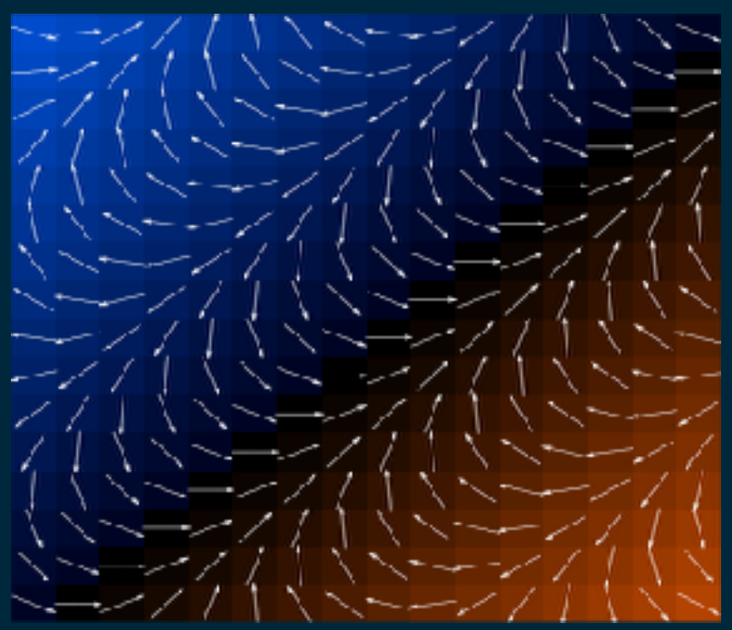
$$M_{rot}(x,t) = \rho(x)e^{-i\gamma G x t}$$

Rotating Frame Magnetisation: **Temporal** Frequency

$$M_{rot}(x_0, t) = \rho(x_0)e^{-i(\gamma G x_0)t} \quad [6a]$$

Rotating Frame Magnetisation: **Spatial** Frequency

$$M_{rot}(x, t_0) = \rho(x)e^{-i(\gamma G t_0)x} \quad [6b]$$



Rotating Frame Magnetisation: **Temporal** Frequency

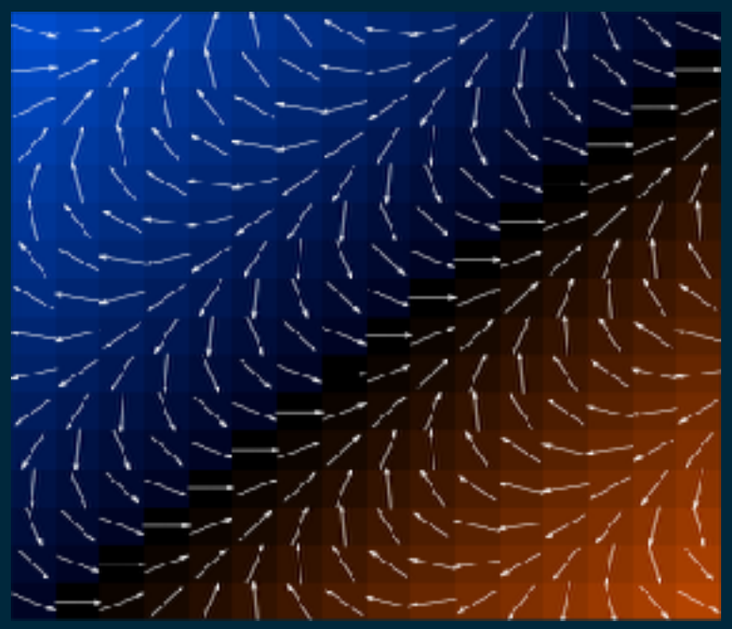
$$M_{rot}(x_0, t) = \rho(x_0)e^{-i\underline{(\gamma G x_0)}t} \quad [6a]$$

Phase rate of change w.r.t. time

$$\frac{d\theta}{dt}$$

Rotating Frame Magnetisation: **Spatial** Frequency

$$M_{rot}(x, t_0) = \rho(x)e^{-i\underline{(\gamma G t_0)}x} \quad [6b]$$



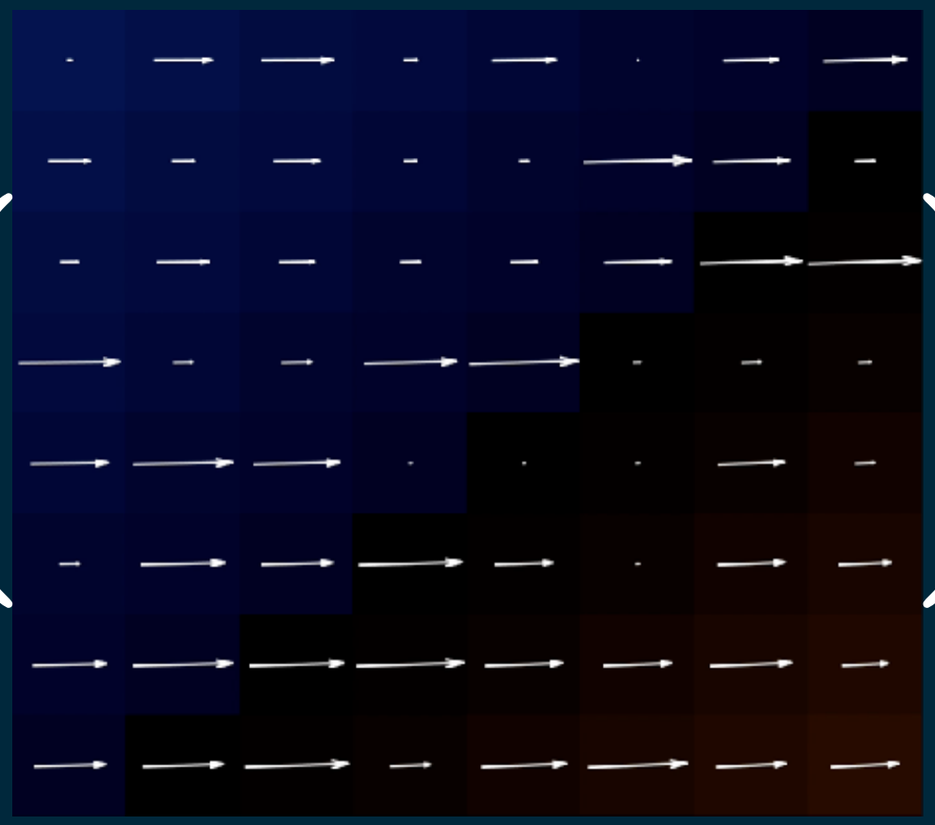
Phase rate of change w.r.t. space

$$\frac{d\theta}{dx}$$

Receive Coils

$$s(t) = \int M_{rot}(x,t)dx \qquad [7]$$

$$= \int \left(\right.$$



$\left. \right)$

Coils integrate signal across all space

Magnetisation

$$M_{rot}(x,t) = \rho(x)e^{-i(\gamma G t)x} \quad [6]$$

Coil Integration

$$s(t) = \int M_{rot}(x,t)dx \quad [7]$$

Signal Equation

$$s(t) = \int \rho(x)e^{-i(\gamma G t)x}dx \quad [8]$$

Signal Equation

$$s(t) = \int \rho(x) e^{-i(\gamma G t)x} dx \qquad [8]$$

Definition [edit]

https://en.wikipedia.org/wiki/Fourier_transform

The Fourier transform of the function f is traditionally den

$f : \mathbb{R} \to \mathbb{C}$.[1][2] Here we will use the following definition:

$$\hat{f}(\xi) = \int_{-\infty}^{\infty} f(x)\, e^{-2\pi i x \xi}\, dx,$$

Signal Equation

$$s(t) = \int \rho(x) e^{-i(\gamma G t)x} dx \qquad [8]$$

Fourier Transform

$$\hat{s}(k) = \int \text{Signal Equation}\ \rho(x) e^{-i(2\pi k)x} dx \qquad [10]$$

Signal Equation

$$s(t) = \int \rho(x) e^{-i(\gamma G t)x} dx \qquad [8]$$

Change of variables

$$k = \frac{\gamma}{2\pi} G t = \frac{\gamma}{2\pi} \int_0 G(\tau) d\tau \qquad [9]$$

Fourier Transform

$$\hat{s}(k) = \int \text{Signal Equation}$$

$$\hat{s}(k) = \int \rho(x) e^{-i(2\pi k)x} dx \qquad [10]$$

Signal Equation

$$s(t) = \int \rho(x) e^{-i(\gamma G t)x} dx \qquad [8]$$

Spatial Frequency variable

$$k = \frac{\gamma}{2\pi} Gt = \frac{\gamma}{2\pi} \int_0^t G(\tau) d\tau \qquad [9]$$

$$\theta = (\gamma G t)x \implies k = \frac{\partial \theta}{\partial x} = \gamma G t$$

Consistent with definition

MR Signal Equation

$$\hat{s}(k) = \int \rho(x) e^{-i(2\pi k)x} dx \qquad [10]$$

Physical Perspective

MR Signal Equation

$$\hat{s}(k) = \int \rho(x) e^{-i(2\pi k)x} dx \qquad [10]$$

What the scanner "measures"

MR Signal Equation

$$\hat{s}(k) = \int \rho(x) e^{-i(2\pi k)x} dx \qquad [10]$$

Magnetisation magnitude at every point in space

MR Signal Equation

$$\hat{s}(k) = \int \rho(x) e^{-i(2\pi k)x} dx \qquad [10]$$

Magnetisation phase at every point in space

Dictated by Bloch equation and linear magnetic fields

Unfortunately we can't make these like "delta" functions

MR Signal Equation

$$\hat{s}(k) = \int \rho(x) e^{-i(2\pi k)x} dx \qquad [10]$$

Coil integrates vector magnetisation over all space

We can't prevent the coil from seeing "everything"

Unfortunately we can't "pick out" specific x-locations

MR Signal Equation

$$\hat{s}(k) = \int \rho(x) e^{-i(2\pi k)x} dx \qquad [10]$$

Fourier Transform Perspective

MR Signal Equation

$$\hat{s}(k) = \int \rho(x) e^{-i(2\pi k)x} dx \qquad [10]$$

Our signal in the “standard” pixel basis:

MR Signal Equation

$$\hat{s}(k) = \int \rho(x) e^{-i(2\pi k)x} dx \qquad [10]$$

New basis of complex exponentials

MR Signal Equation

$$\hat{s}(k) = \int \rho(x) e^{-i(2\pi k)x} dx \qquad [10]$$

Inner product to compute Fourier coefficients

i.e. change of basis from standard to Fourier

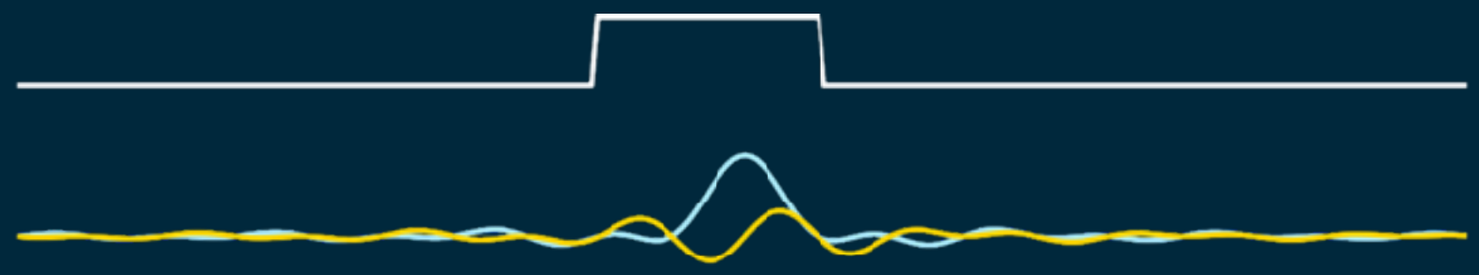
MR Signal Equation

$$\hat{s}(k) = \int \rho(x) e^{-i(2\pi k)x} dx \qquad [10]$$

Same signal in new (Fourier) basis

MR Signal Equation

$$\hat{s}(k) = \int \rho(x) e^{-i(2\pi k)x} dx \qquad [10]$$



Discretised Signal Equation

$$\hat{s}(k) = \sum_{n=0}^{N-1} \rho(x_n) e^{-i2\pi kn/N} \qquad [11]$$

Discretised
Signal Equation

$$\hat{s}(k) = \sum_{n=0}^{N-1} \rho(x_n) e^{-i2\pi kn/N} \qquad [11]$$

$$\begin{bmatrix} \hat{s}(k_0) \\ \hat{s}(k_1) \\ \vdots \\ \hat{s}(k_{N-1}) \end{bmatrix} = \begin{bmatrix} e^{-i2\pi k_0(0)/N} & e^{-i2\pi k_0(1)/N} & \dots & e^{-i2\pi k_0(N-1)/N} \\ e^{-i2\pi k_1(0)/N} & e^{-i2\pi k_1(1)/N} & \dots & e^{-i2\pi k_1(N-1)/N} \\ \vdots & \vdots & \vdots & \vdots \\ e^{-i2\pi k_{N-1}(0)/N} & e^{-i2\pi k_{N-1}(1)/N} & \dots & e^{-i2\pi k_{N-1}(N-1)/N} \end{bmatrix} \begin{bmatrix} \rho(x_0) \\ \rho(x_1) \\ \vdots \\ \rho(x_{N-1}) \end{bmatrix}$$

Matrix Signal Equation

$$\hat{s} = F\rho \qquad [12]$$

$$\begin{bmatrix} \hat{s}(k_0) \\ \hat{s}(k_1) \\ \vdots \\ \hat{s}(k_{N-1}) \end{bmatrix} = \begin{bmatrix} e^{-i2\pi k_0(0)/N} & e^{-i2\pi k_0(1)/N} & \dots & e^{-i2\pi k_0(N-1)/N} \\ e^{-i2\pi k_1(0)/N} & e^{-i2\pi k_1(1)/N} & \dots & e^{-i2\pi k_1(N-1)/N} \\ \vdots & \vdots & \vdots & \vdots \\ e^{-i2\pi k_{N-1}(0)/N} & e^{-i2\pi k_{N-1}(1)/N} & \dots & e^{-i2\pi k_{N-1}(N-1)/N} \end{bmatrix} \begin{bmatrix} \rho(x_0) \\ \rho(x_1) \\ \vdots \\ \rho(x_{N-1}) \end{bmatrix}$$

To construct each measurement:

Fourier Encoding Matrix *rotates* the magnetisation
on a voxel-by-voxel basis

This is *physically* enforced by the magnetic field gradients

Then the coil sums/integrates all the rotated voxel signals

$$\begin{bmatrix} \hat{s}(k_0) \\ \hat{s}(k_1) \\ \vdots \\ \hat{s}(k_{N-1}) \end{bmatrix} = \begin{bmatrix} e^{-i2\pi k_0(0)/N} & e^{-i2\pi k_0(1)/N} & \dots & e^{-i2\pi k_0(N-1)/N} \\ e^{-i2\pi k_1(0)/N} & e^{-i2\pi k_1(1)/N} & \dots & e^{-i2\pi k_1(N-1)/N} \\ \vdots & \vdots & \vdots & \vdots \\ e^{-i2\pi k_{N-1}(0)/N} & e^{-i2\pi k_{N-1}(1)/N} & \dots & e^{-i2\pi k_{N-1}(N-1)/N} \end{bmatrix} \begin{bmatrix} \rho(x_0) \\ \rho(x_1) \\ \vdots \\ \rho(x_{N-1}) \end{bmatrix}$$

To construct each measurement:

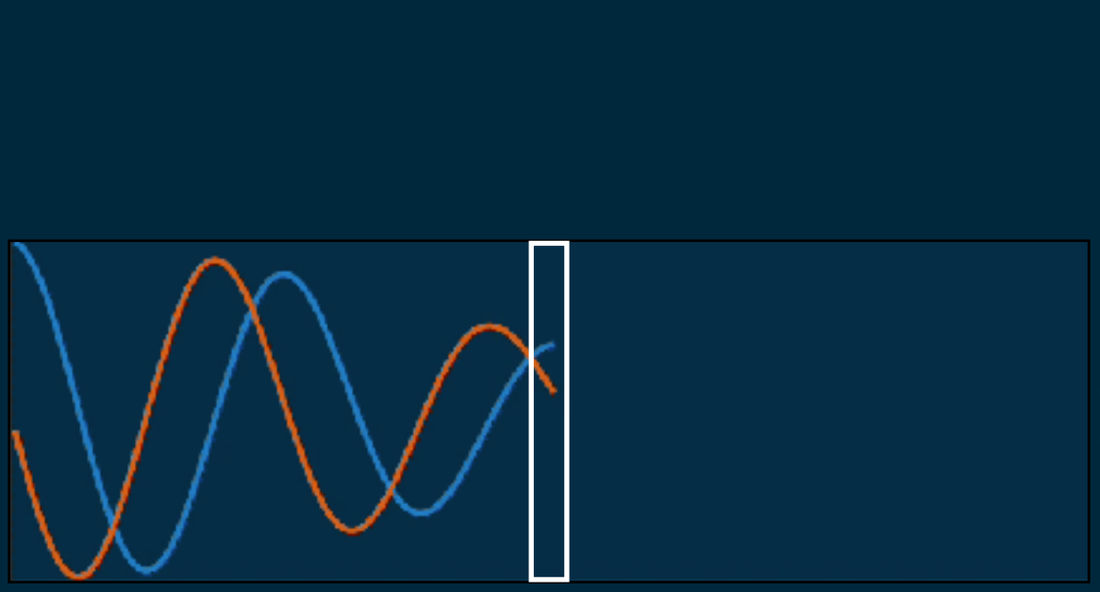
Fourier Encoding Matrix *rotates* the magnetisation
on a voxel-by-voxel basis

The *phase* of the magnetisation is manipulated
across space to *encode* information about the image

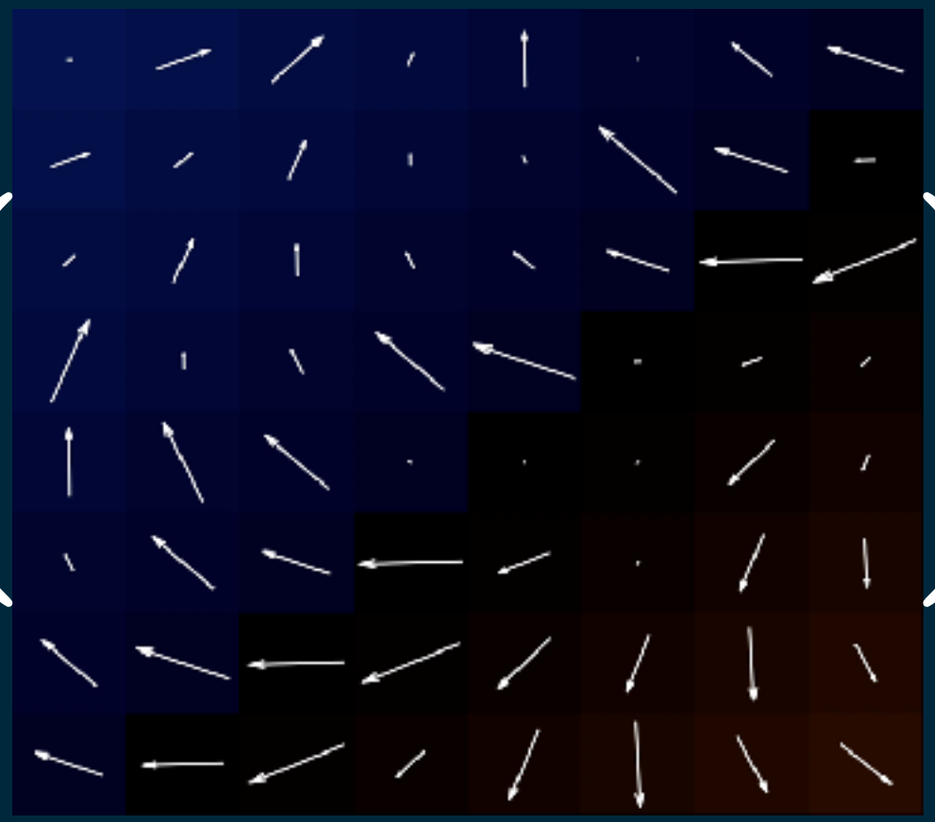
Phase Encoding QED

Sampling is effectively "instantaneous"

Scanner doesn't really *care* at all how fast any given voxel is rotating



$$=\int$$



When I choose to take my sample (measurement) the *only* important quantity is the relative encoded *phases* at that instant

The *fundamental* role of *frequency* and *time* in the encoding process (ignoring signal decay)

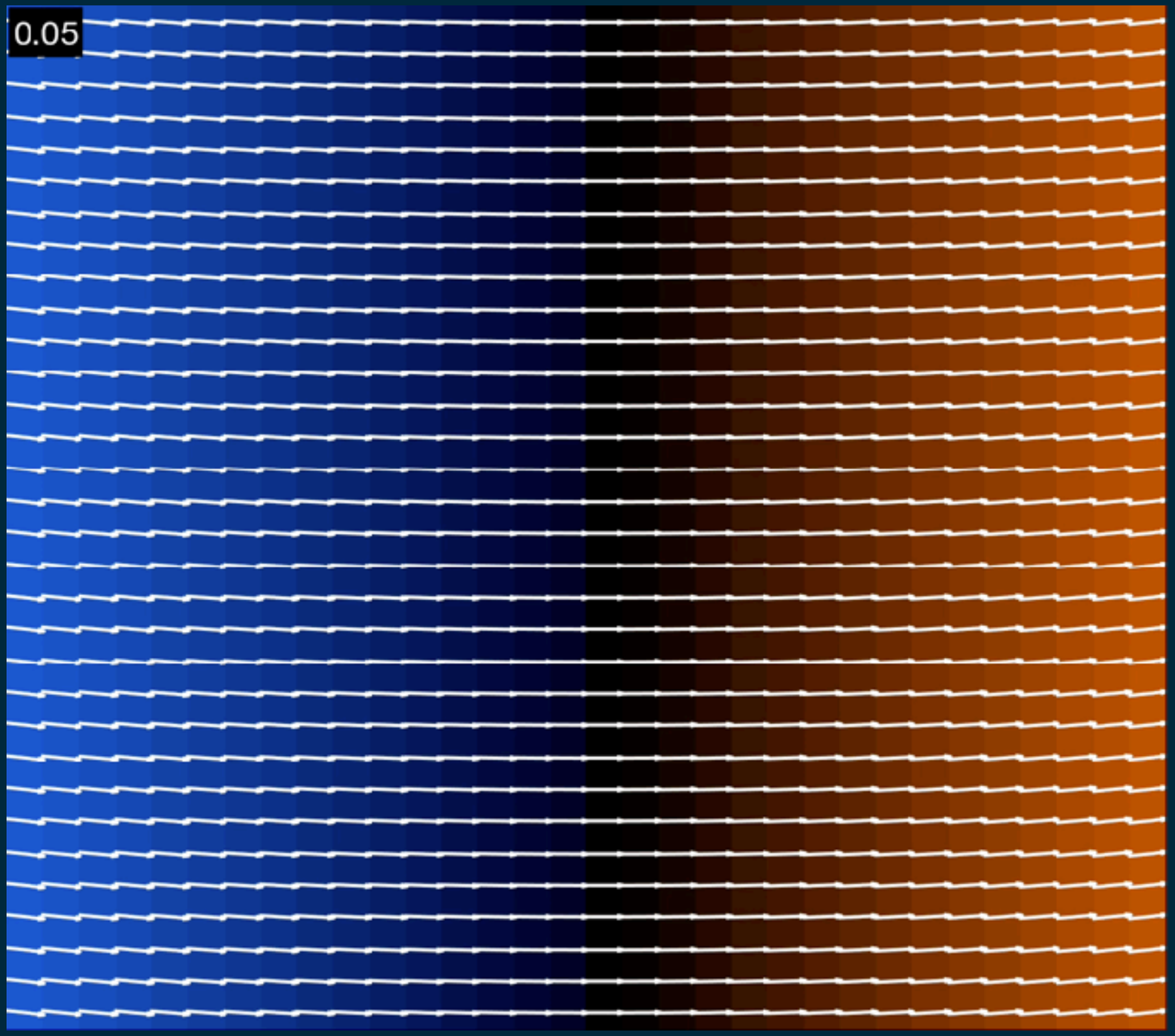
Is only in facilitating different phase encoding distributions across space

$$k = \frac{\gamma}{2\pi} Gt = \frac{\gamma}{2\pi} \int_0^t G(\tau) d\tau \qquad [9]$$

Time and *frequency* are the means

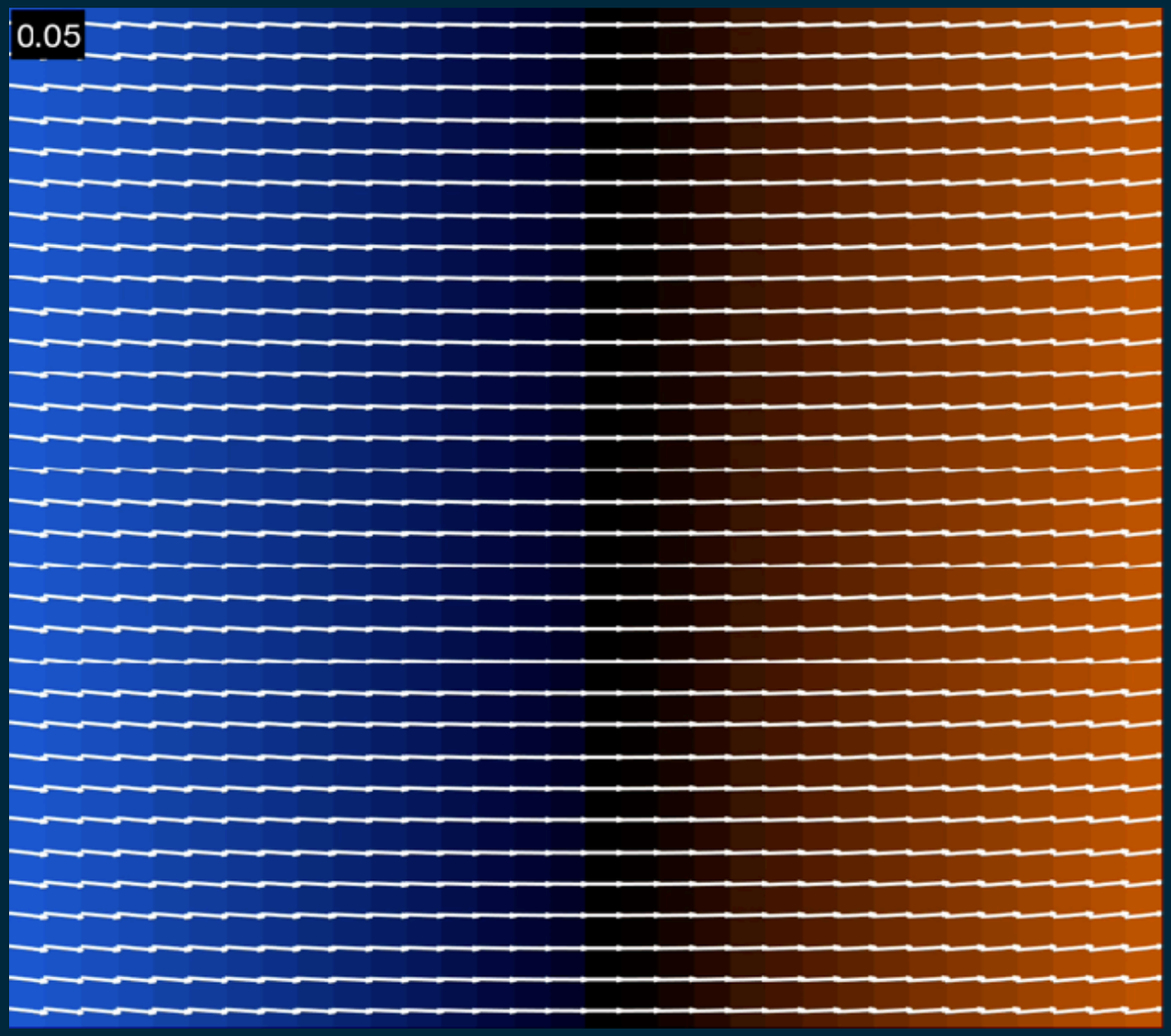
Spatially-varying phase is the end

The "action" of an x-gradient over *time*

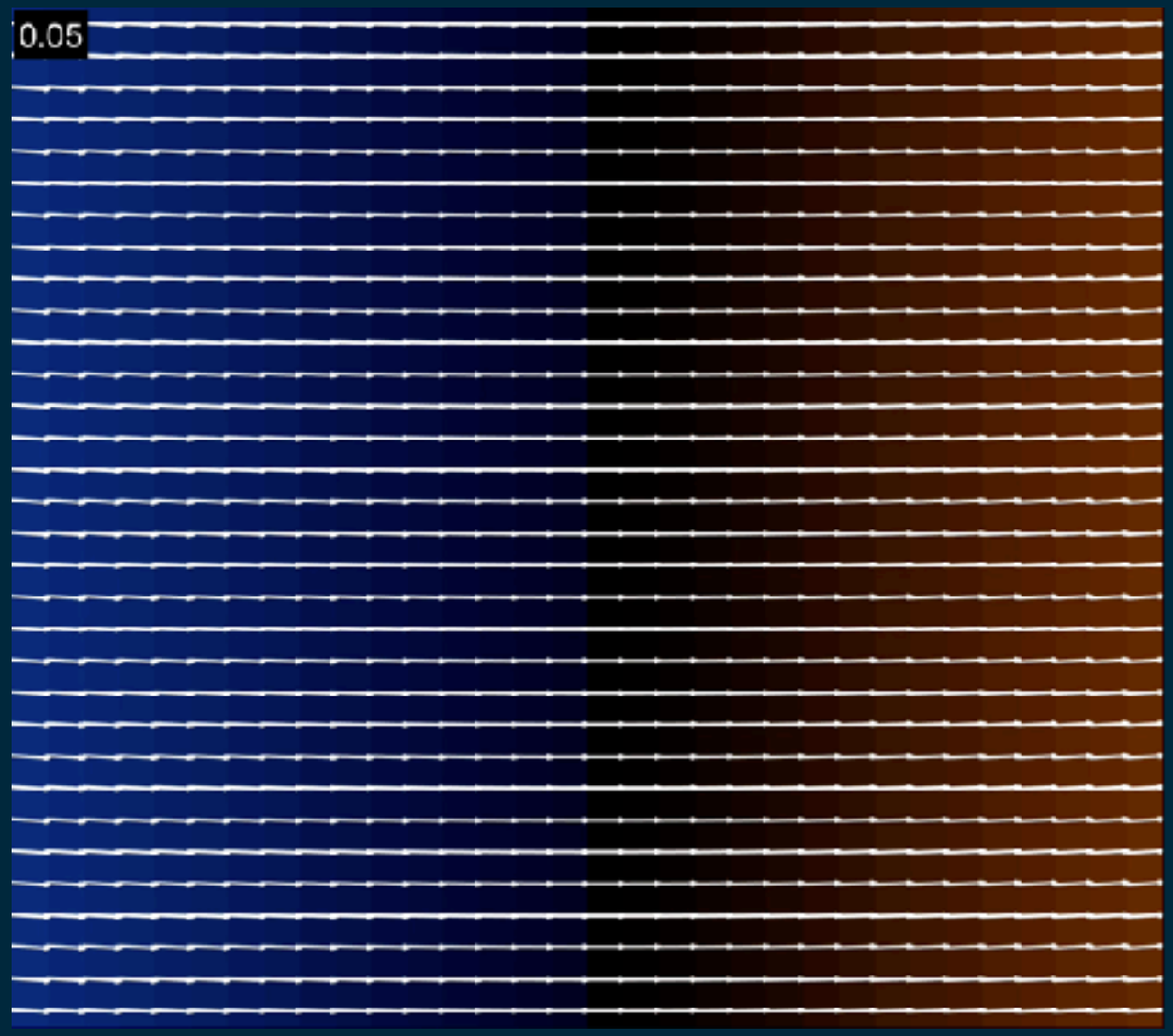


2 different strength x-gradients

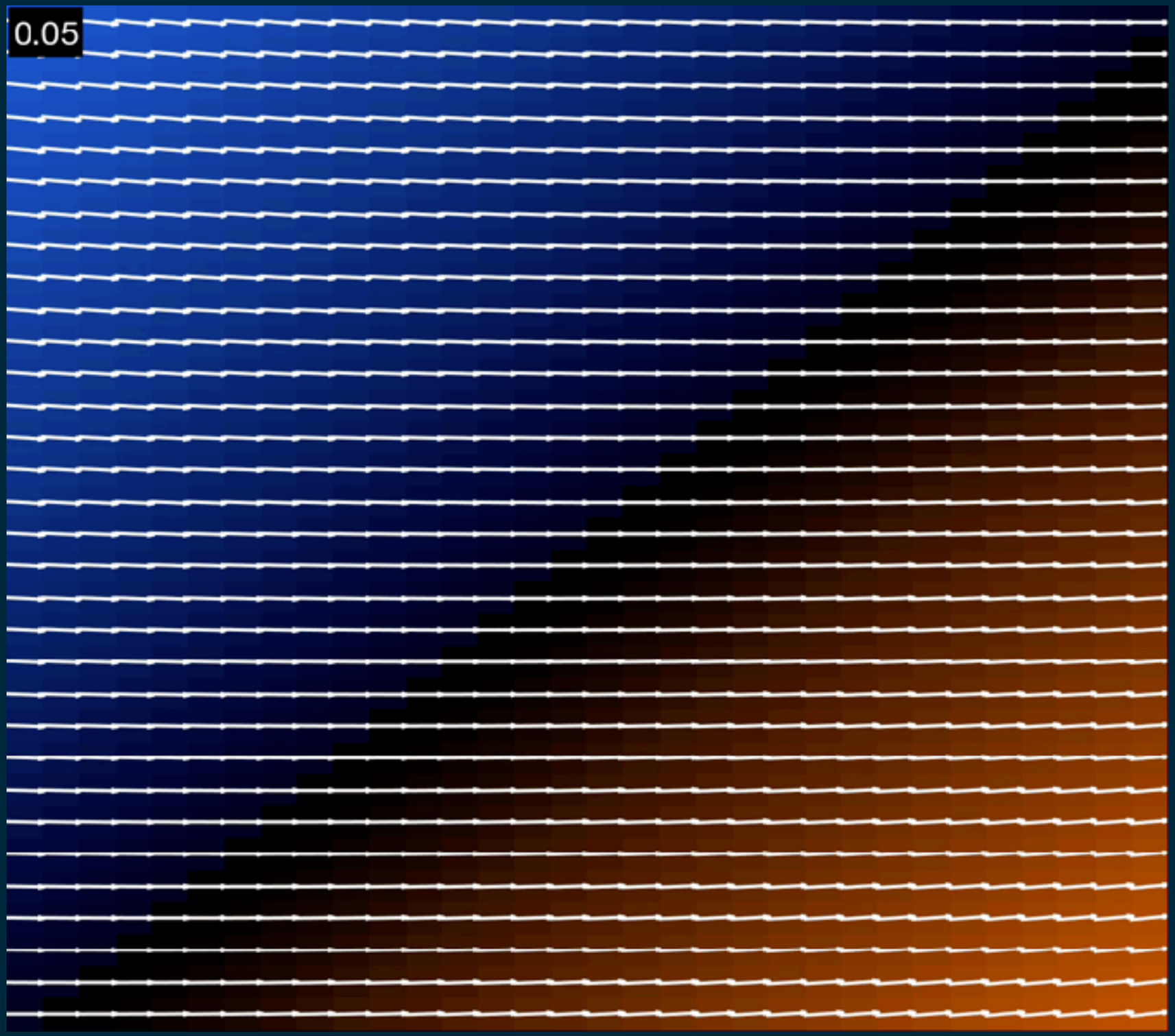
G_x



$\frac{1}{2}G_x$



The “action” of a x+y gradient over *time*



$$k = \frac{\gamma}{2\pi} Gt = \frac{\gamma}{2\pi} \int_0^t G(\tau) d\tau \qquad [9]$$

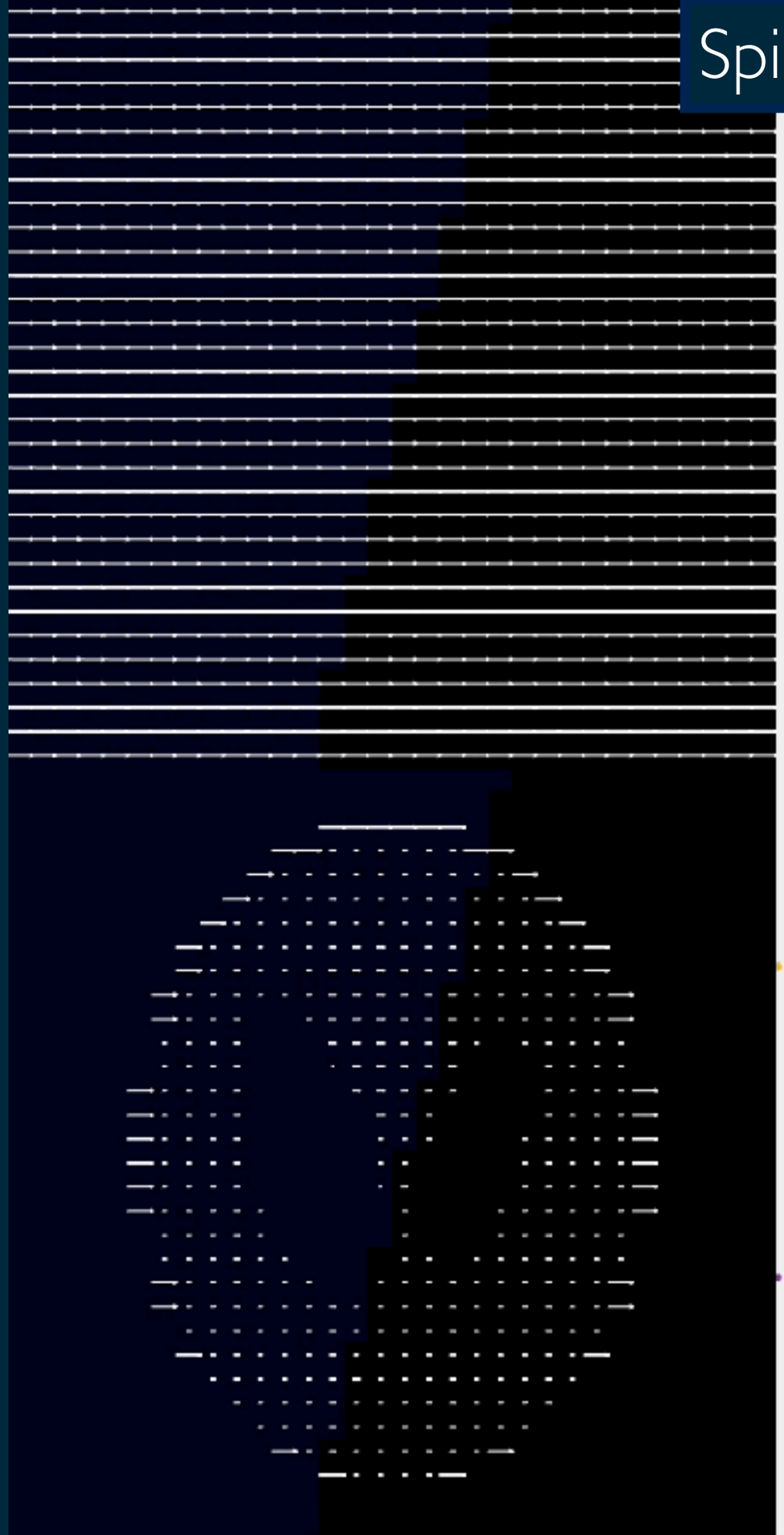
The *k* variable represents the *location* in Fourier space

Different *locations* in Fourier space represent different *phase encoding* patterns (i.e. *spatial* frequencies)

We need to acquire samples on *at least* N linearly-independent *k-space locations (k-values)* to fully sample our image

Spiral

Encoding Phase



k-position and signal magnitude

Object with encoding

Real + Imag signals

Encoding Phase

EPI

k-position and signal magnitude

Object with encoding

Real + Imag signals

The Imaging Point-Spread Function Governs MR Sampling

The Point-Spread Function is an Impulse Response

Linear Shift Invariant Systems can be characterised by their Impulse Response

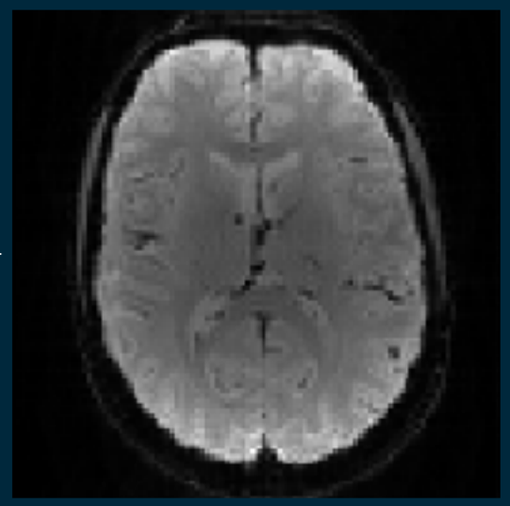
MR Imaging is Linear

Scaling 

Superposition 

MR Imaging is Shift Invariant

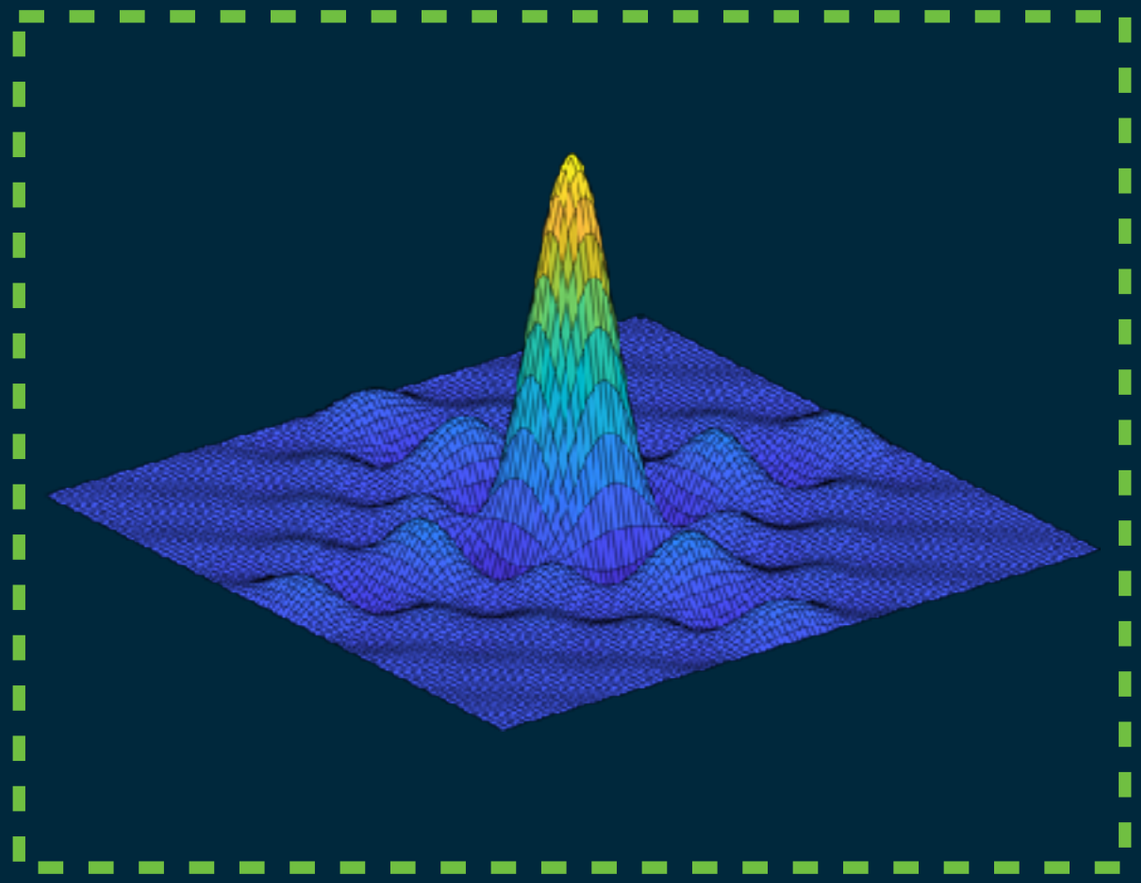
Linear Shift Invariant

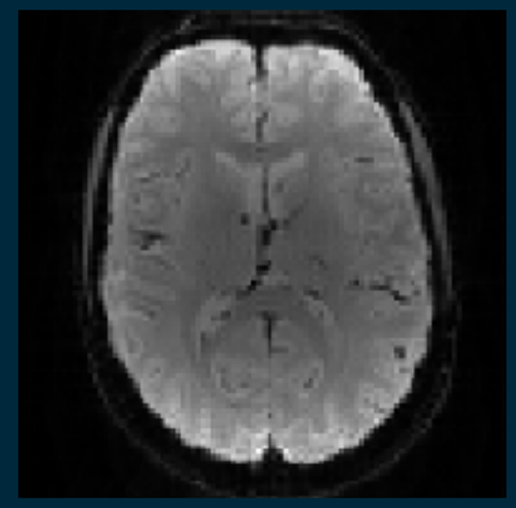


Has a point spread function

What is the point-spread function (PSF)?

The action of MR sampling can be replaced





by the convolution of a PSF

How do we compute the PSF?

The PSF is the impulse response

Compute the response to a point-source

$$PSF(x) = (\mathcal{F}^{-1}\Psi\mathcal{F})\delta(x)$$

$$PSF(x) = (\mathcal{F}^{-1}\Psi\mathcal{F})\delta(x)$$

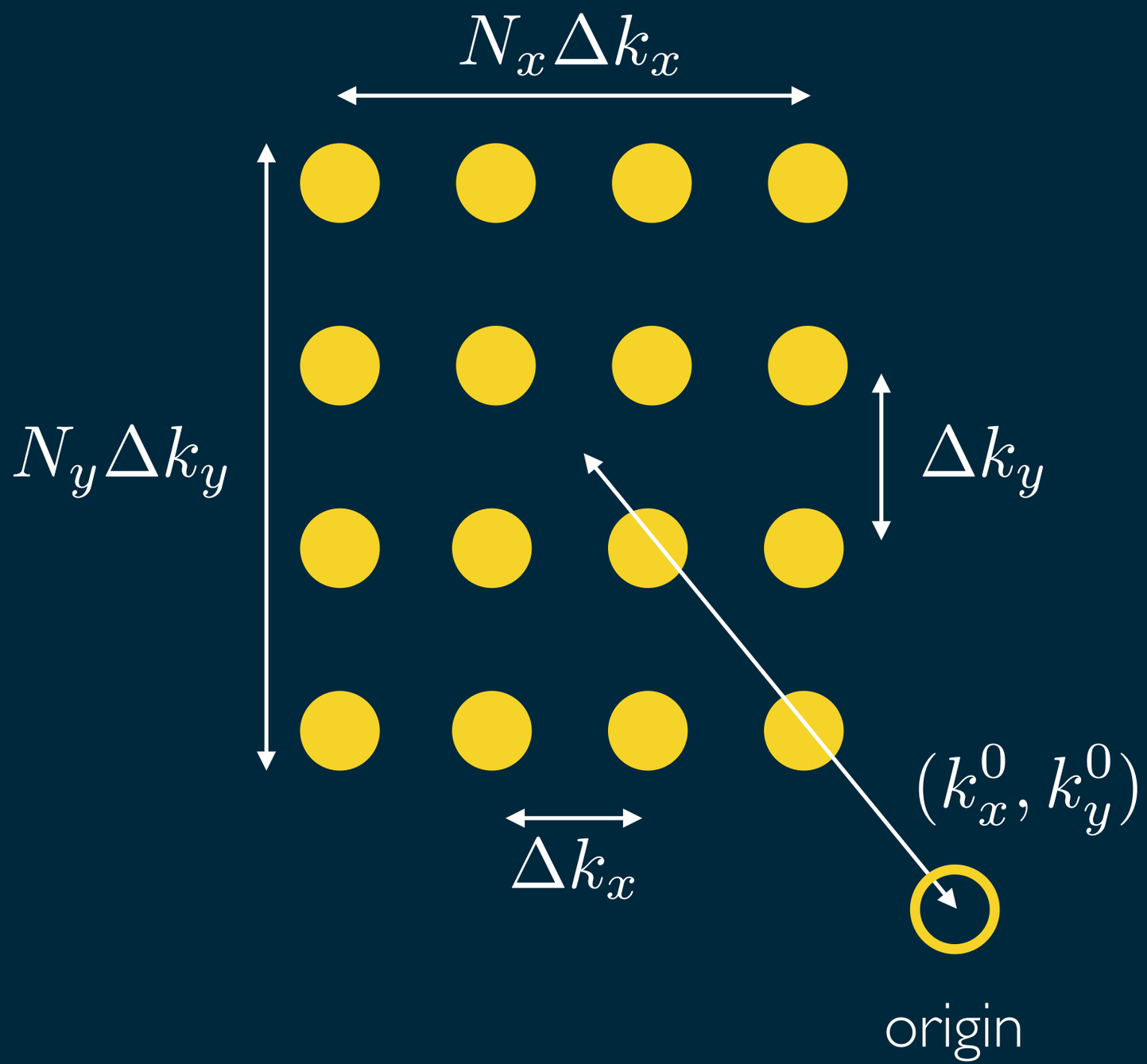
The Fourier Transform of a point (delta function) is constant

Sample and inverse transform a uniform (all ones) k-space

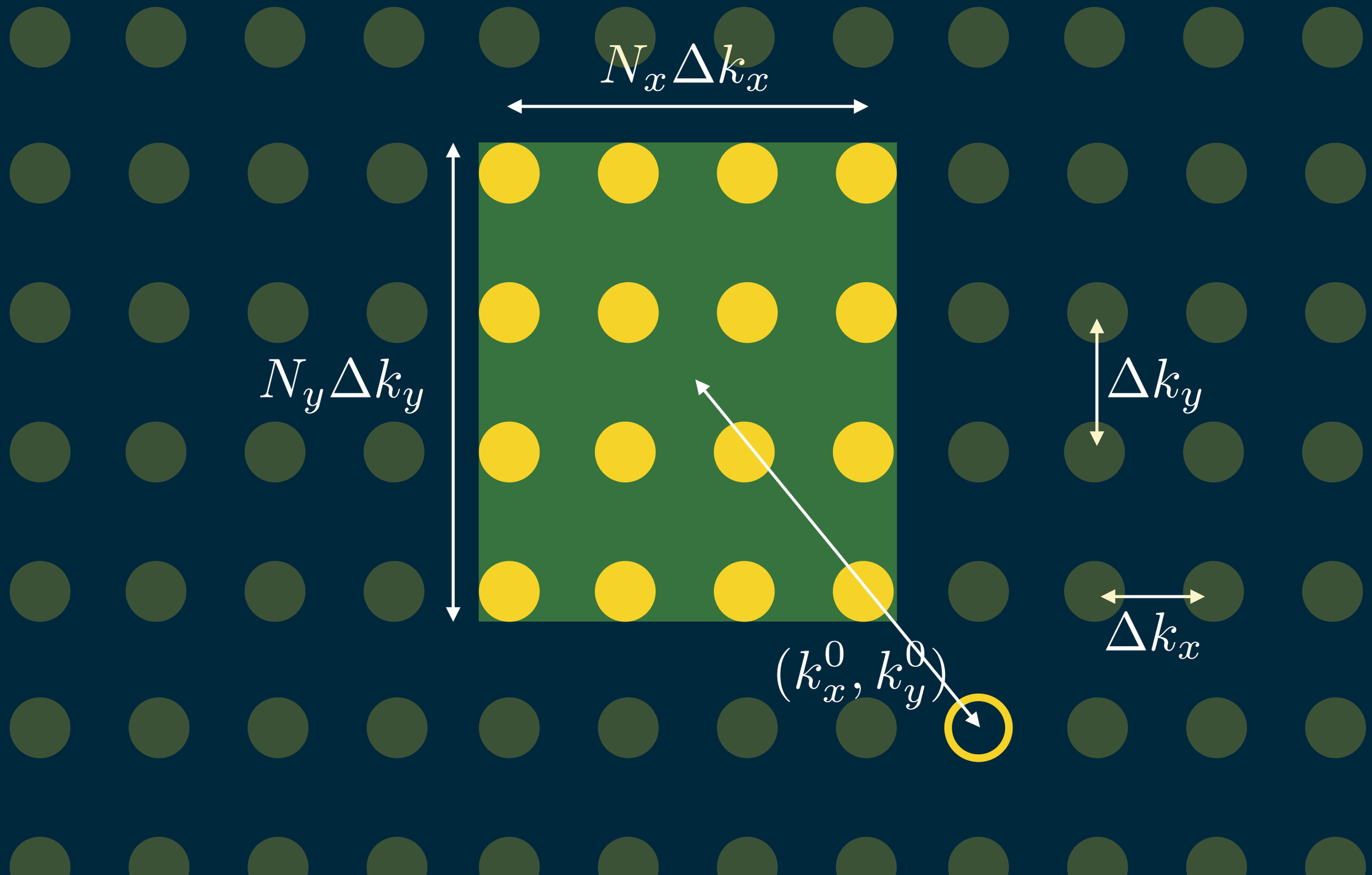
$$PSF(x) = \mathcal{F}^{-1}\Psi$$

The PSF characterises our k-space sampling choices

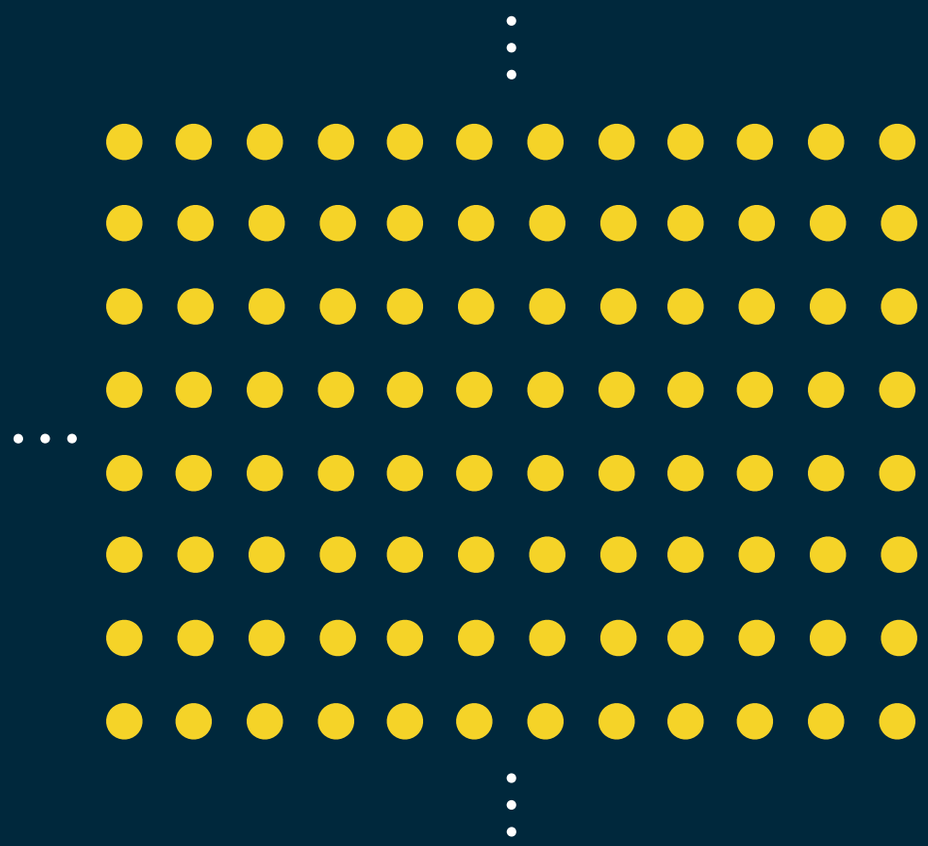
Cartesian Sampling of k-space



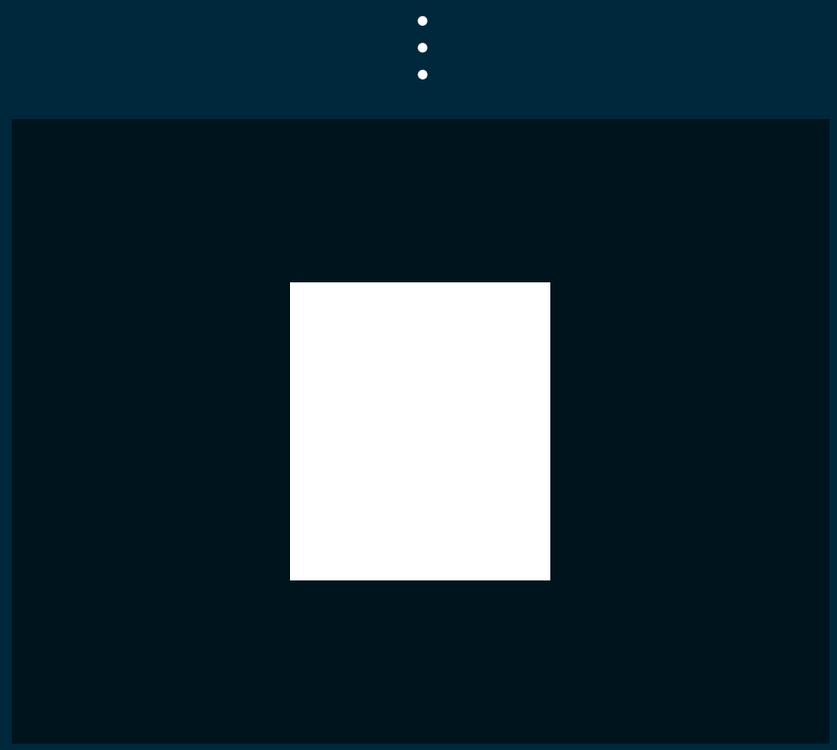
Cartesian Sampling of k-space



Dirac Comb Function



Rectangular Window



$$\text{III}(k_x, k_y) = \sum_{m,n=-\infty}^{\infty} \delta(k_x - m\Delta k_x, k_y - n\Delta k_y)$$

$$rect(k_x) = \begin{cases} 1 & \frac{-N_x \Delta k_x}{2} < (k_x - k_x^0) < \frac{N_x \Delta k_x}{2} \\ 0 & otherwise \end{cases}$$

$$rect(k_y) = \begin{cases} 1 & \frac{-N_y \Delta k_y}{2} < (k_y - k_y^0) < \frac{N_y \Delta k_y}{2} \\ 0 & otherwise \end{cases}$$

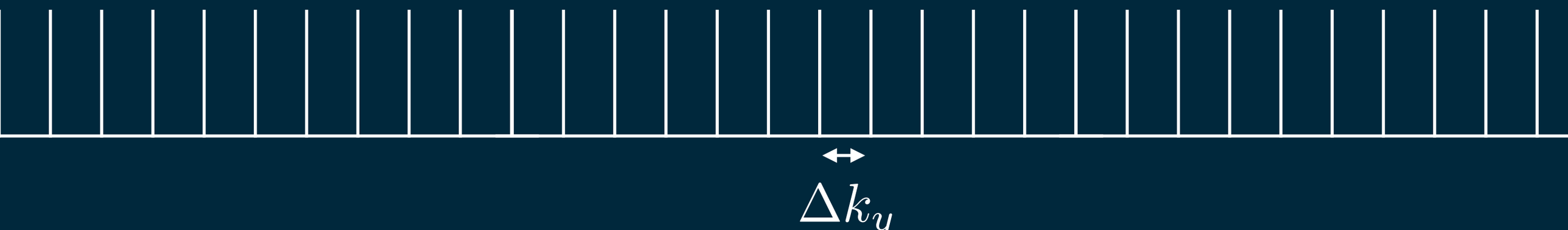
$$\Psi(k_x, k_y) = \mathrm{III}(k_x, k_y) \cdot \mathrm{rect}(k_x) \cdot \mathrm{rect}(k_y)$$

This is a convenient representation because
their inverse Fourier Transforms are well known

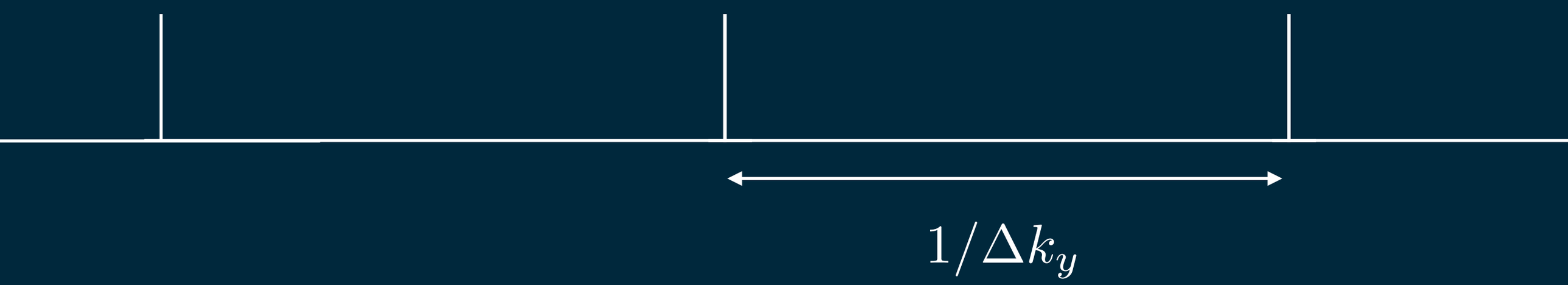
Also easy to compute numerically as the inverse Fourier transform
of an "all ones" k-space

$$PSF(x, y) = \mathcal{F}^{-1}\left(\mathrm{III}(k_x, k_y)\right) \circledast \mathcal{F}^{-1}\left(\mathrm{rect}(k_x)\mathrm{rect}(k_y)\right)$$

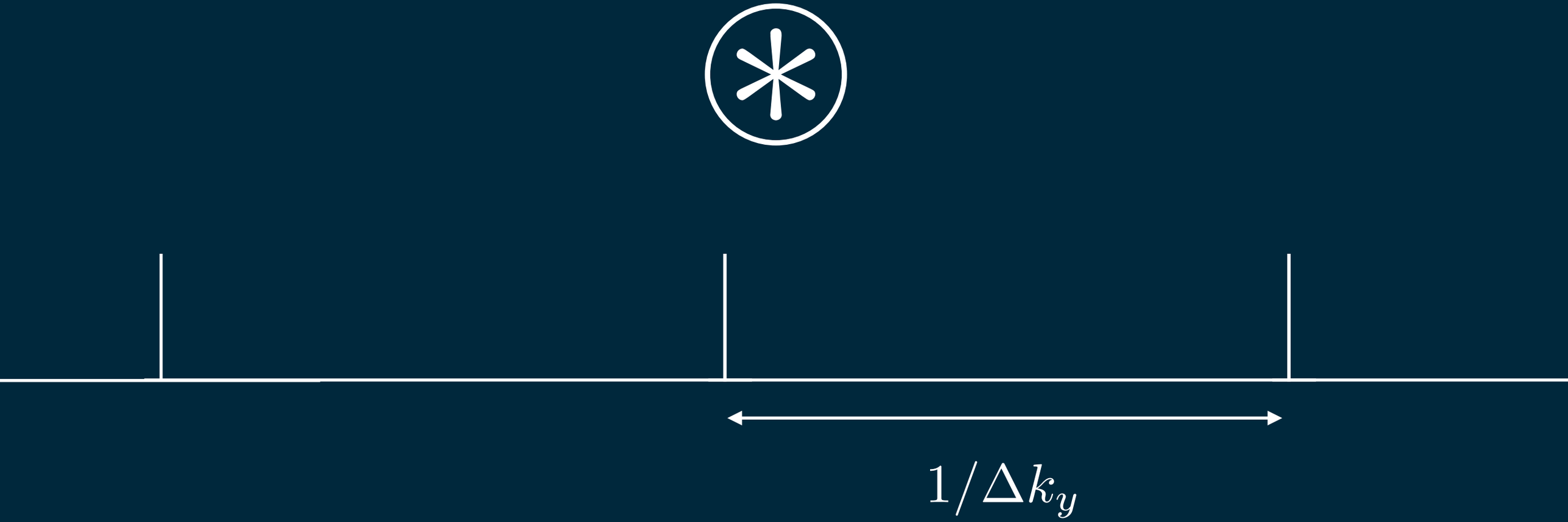
1D Dirac Comb Function



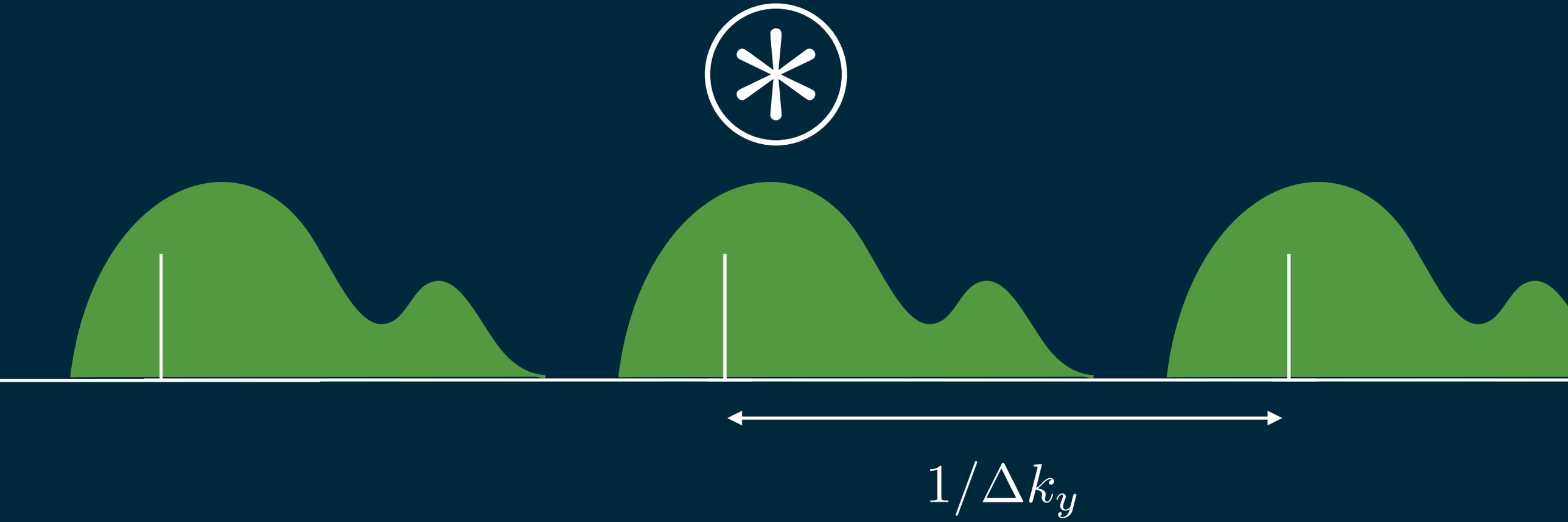
$$\mathcal{F}^{-1}\left(\sum_{n=-\infty}^{\infty}\delta(k_y - n\Delta k_y)\right) \propto \sum_{n=-\infty}^{\infty}\delta(y - \frac{n}{\Delta k_y})$$



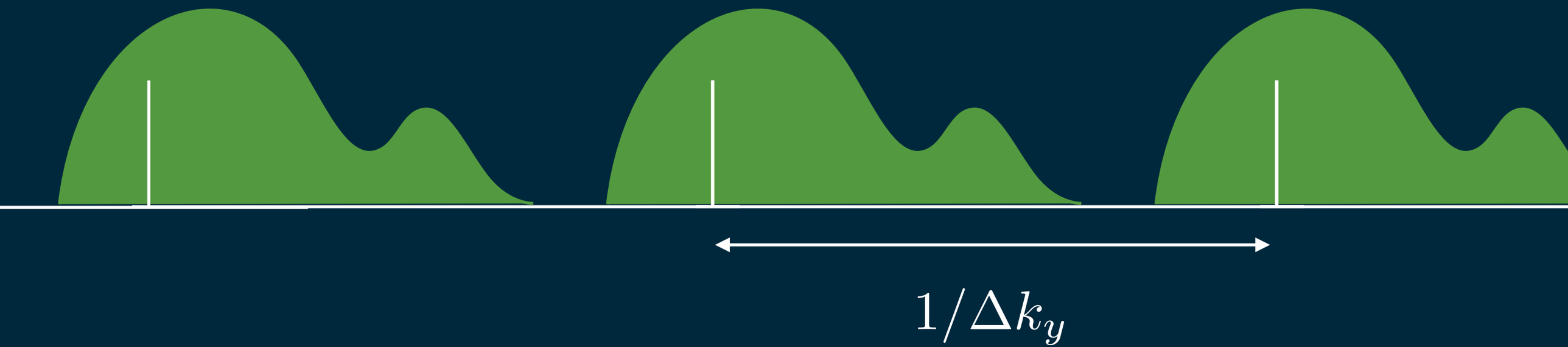
1D Dirac Comb Function



1D Dirac Comb Function



$1/\Delta k_y$ dictates the spacing between copies

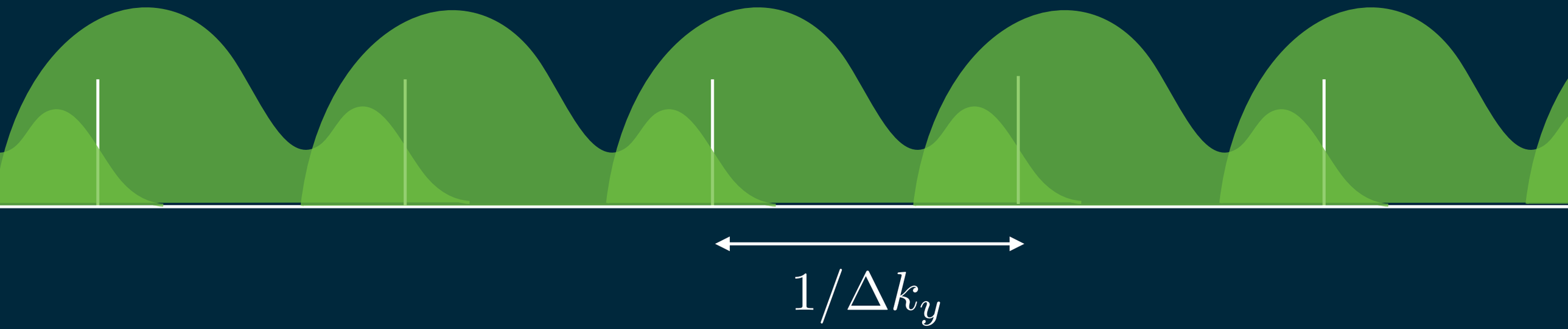


it corresponds to the maximum width of the object without aliasing

it also represents the largest possible window with unique information

$$1/\Delta k_y = FOV_y$$

Field of View

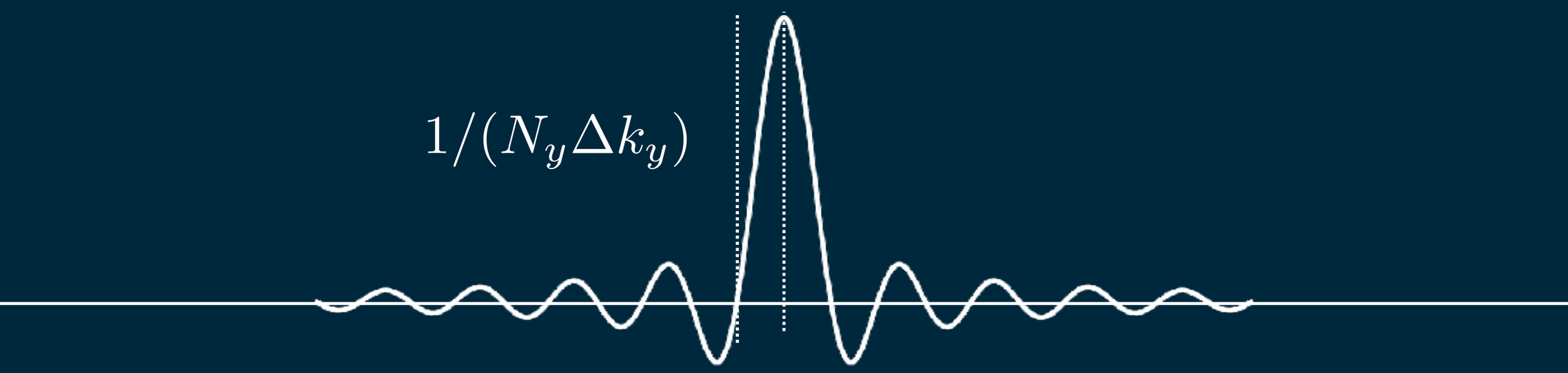


When it is too small, you get aliasing/overlap

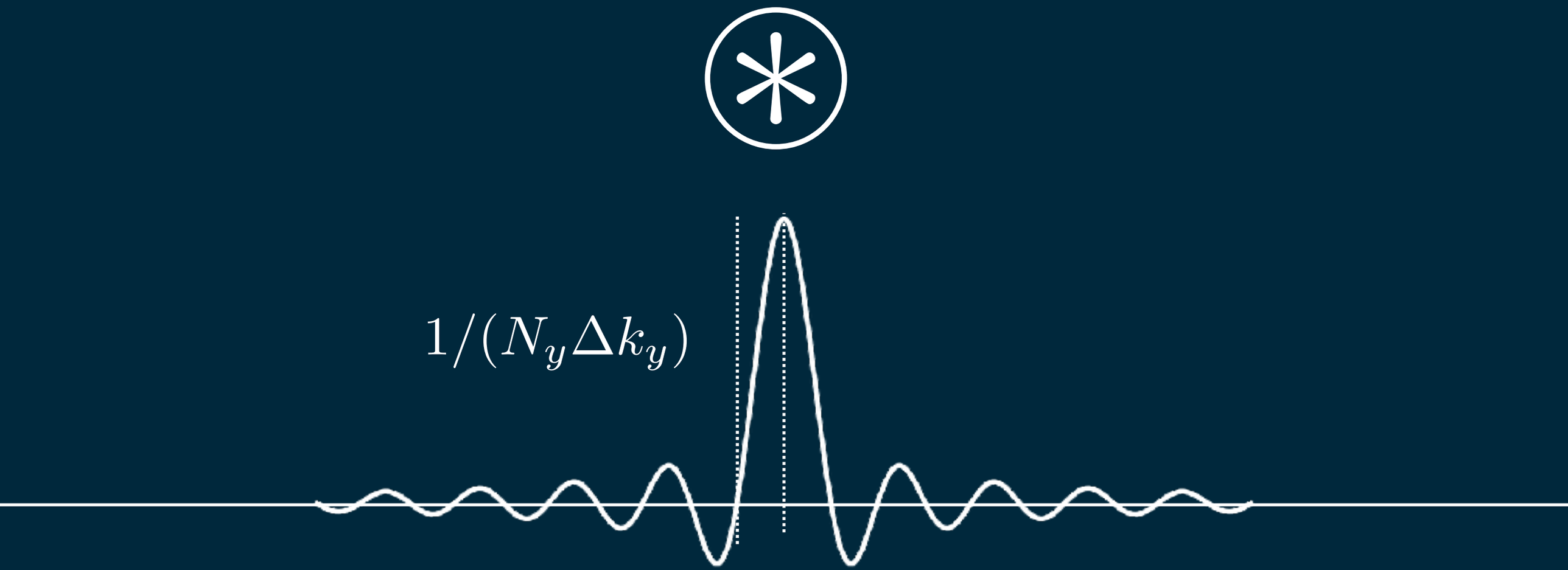
They result from successive (infinite) copies being too close together

1D Rectangular Window

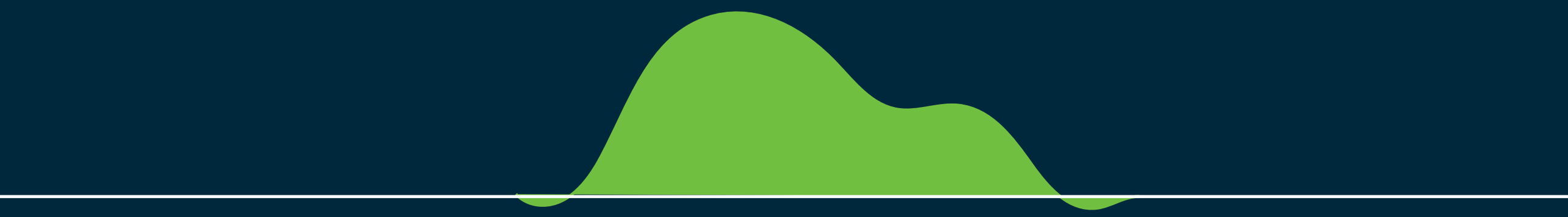
$$\mathcal{F}^{-1}\left(\text{rect}\left(\frac{k_y - k_y^0}{N_y \Delta k_y}\right)\right) \propto e^{-i2\pi k_y^0 y} \cdot \text{sinc}(N_y \Delta k_y y)$$



1D Rectangular Window



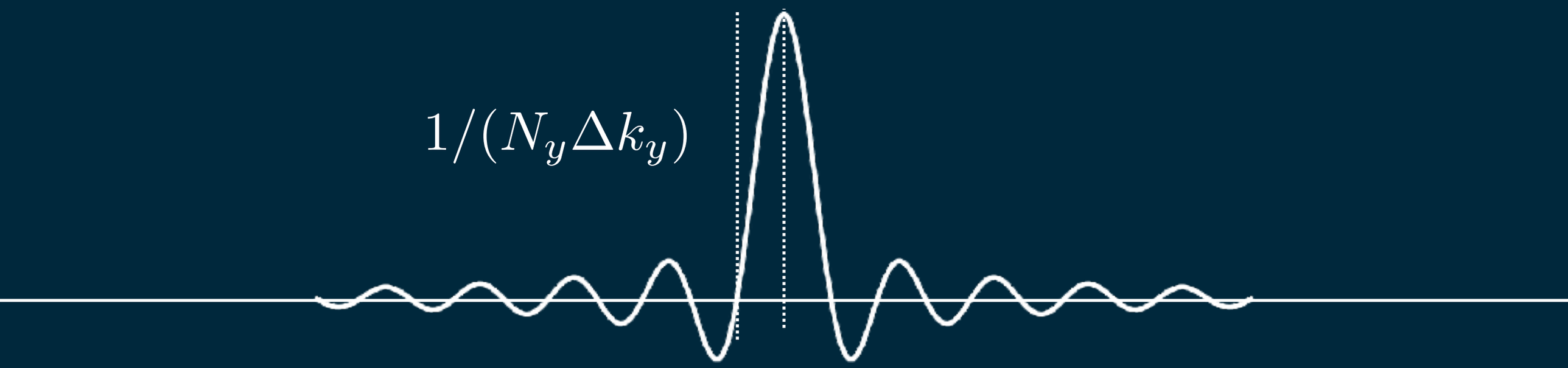
Spatial Resolution



$$1/(N_y \Delta k_y) = \Delta y$$

Spatial resolution is dictated by the width of the Sinc
In this case, we *choose* the first zero-crossing

In general, this is the width of the inverse Fourier Transform of whatever window function you use

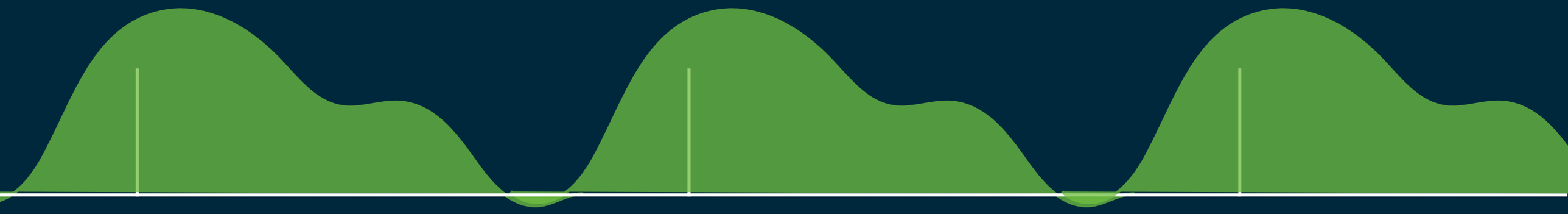


Voxels are *not* rectangles of width (Δx,Δy)

The PSF dictates the *shape* of the voxel

i.e., the contribution across all space of signals that "are" the voxel

The final "image"

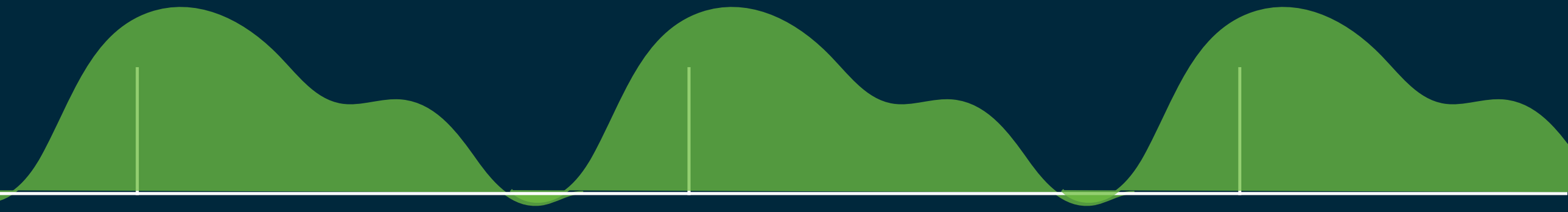


$$\mathcal{F}^{-1}\left[\left(\sum_{n=-\infty}^{\infty}\delta(k_y - n\Delta k_y)\right)\text{rect}\left(\frac{k_y - k_y^0}{N_y\Delta k_y}\right)(\mathcal{F}\rho(y))\right]$$

$$\int\left(\sum_{n=-\infty}^{\infty}\delta(k_y - n\Delta k_y)\right)\text{rect}\left(\frac{k_y - k_y^0}{N_y\Delta k_y}\right)\rho_k(k_y)e^{i2\pi k_y y}dy$$

$$\sum_{n=-N_y/2}^{N_y/2}\rho_k(n\Delta k_y)e^{i2\pi n\Delta k_y y}dy$$

The final "image"



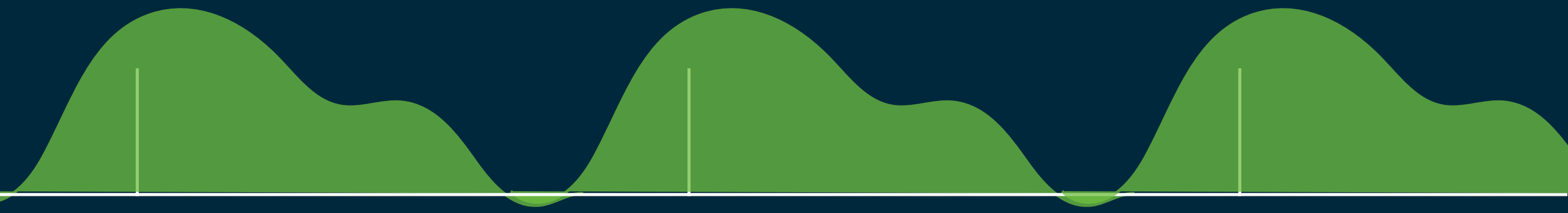
$$\sum_{n=-N_y/2}^{N_y/2} \rho_k(n\Delta k_y) e^{i2\pi n \Delta k_y y}$$

We can evaluate this for *any* y

Our "image" is well defined over *all space*

So what defines our voxel locations?

Inverse Discrete Fourier Transform

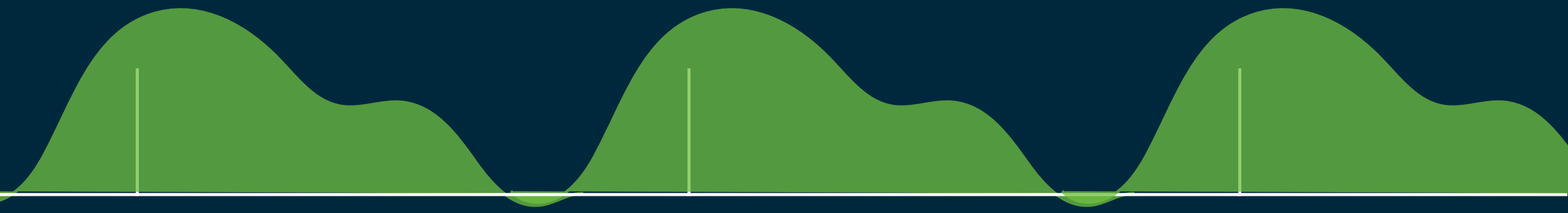


$$\sum_{n=-N_y/2}^{N_y/2} \rho_k(n\Delta k_y) e^{i2\pi n \Delta k_y y}$$

$$\frac{1}{N}\sum_{k=0}^{N-1} X_k \cdot e^{i2\pi kn/N}, \quad n \in \mathbb{Z},$$

https://en.wikipedia.org/wiki/Discrete_Fourier_transform

Inverse Discrete Fourier Transform

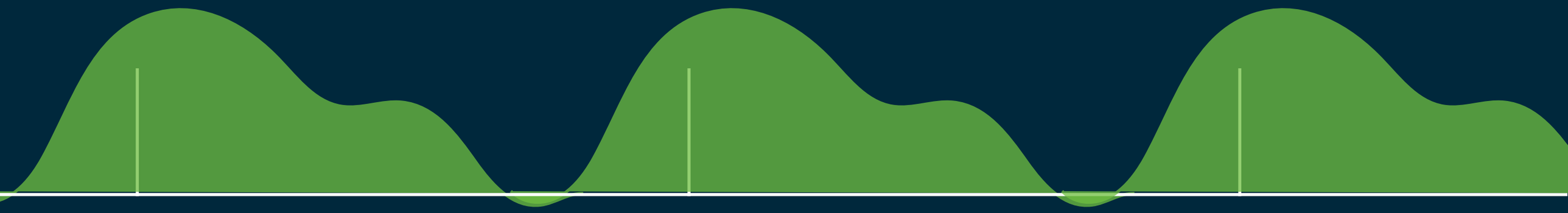


$$\sum_{n=-N_y/2}^{N_y/2} \rho_k(n\Delta k_y) \cdot e^{i2\pi ny/FOV_y}$$

$$\frac{1}{N}\sum_{k=0}^{N-1} X_k \cdot e^{i2\pi kn/N}, \quad n \in \mathbb{Z},$$

https://en.wikipedia.org/wiki/Discrete_Fourier_transform

Inverse Discrete Fourier Transform

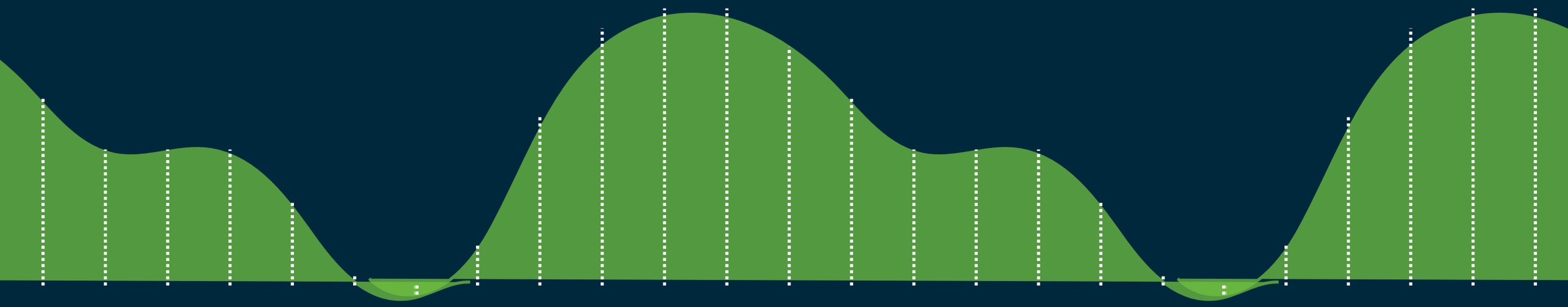


$$\sum_{n=-N_y/2}^{N_y/2} \rho_k(n\Delta k_y) \cdot e^{i2\pi ny/FOV_y}$$

$$\rho(y) = \mathrm{DFT}^{-1}(\rho_k(n\Delta k_y))$$

```
img = ifft(kdata)
```

Inverse Discrete Fourier Transform



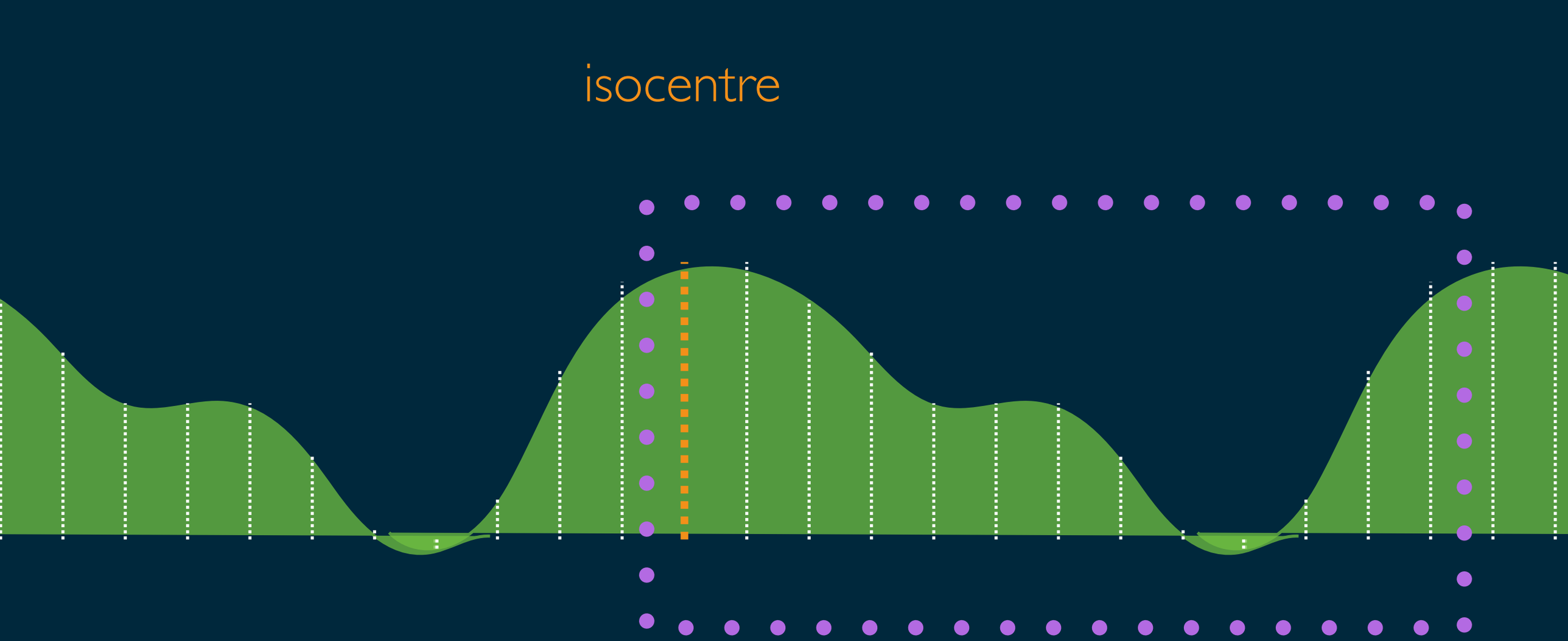
$$\frac{1}{N}\sum_{k=0}^{N-1} X_k \cdot e^{i2\pi kn/N}, \qquad n \in \mathbb{Z},$$

https://en.wikipedia.org/wiki/Discrete_Fourier_transform

Voxel locations are *implicitly* chosen when you select a DFT to use

You can choose non-integer n to get "in between" values

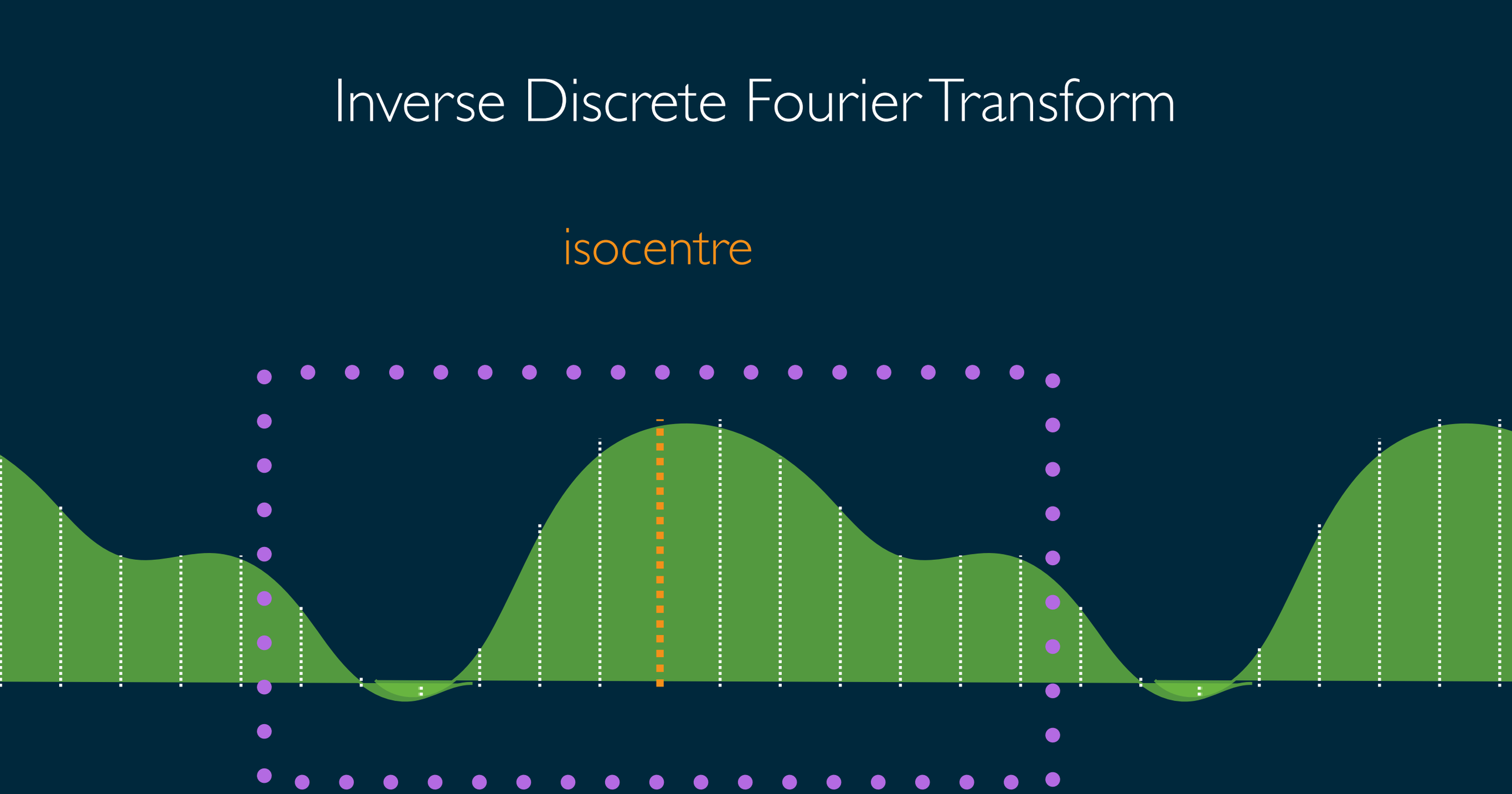
Inverse Discrete Fourier Transform



MATLAB convention

$$n = 0, 1, ..., N - 1$$

Inverse Discrete Fourier Transform



MATLAB after fft-shifting

$$n = -\frac{N}{2}, -\frac{N}{2}+1, \ldots \frac{N}{2}-1$$