

RF Pulse Design

Multi-dimensional Excitation II

Matlab Exercise

M229 Advanced Topics in MRI

Kyung Sung, Ph.D.

2020.04.21

Class Business

- Homework 2 will be out today (due on 5/8)
- Jiahao Lin is TA for HW2

Today's Topics

- Excitation k-space interpretation
- 2D EPI pulse design
- MATLAB exercise

Excitation k-space Interpretation

Small Tip Approximation

$$M_{xy}(t, z) = i\gamma M_0 \int_0^t B_1(s) e^{-i\omega(z)(t-s)} ds$$

$$\omega(z) = \gamma G_z z \quad \longrightarrow \quad \omega(\vec{r}, t) = \gamma \vec{G}(t) \vec{r}$$

$$M_{xy}(t, \vec{r}) = i\gamma M_0 \int_0^t B_1(s) e^{-i\gamma \int_s^t \vec{G}(\tau) d\tau \cdot \vec{r}} ds$$

Small Tip Approximation

$$M_{xy}(t, \vec{r}) = i\gamma M_0 \int_0^t B_1(s) e^{-i\gamma \int_s^t \vec{G}(\tau) d\tau \cdot \vec{r}} ds$$

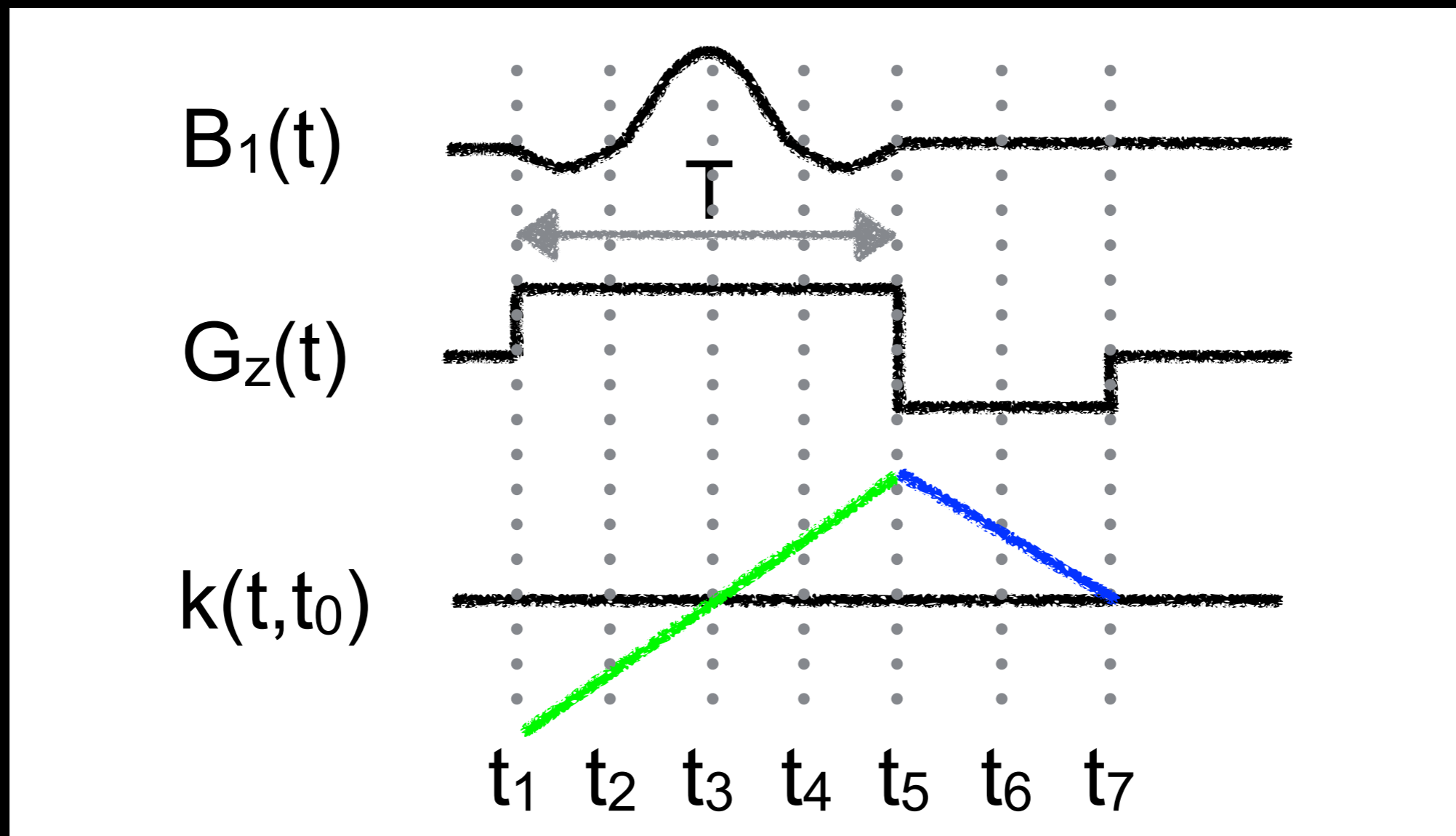
Let us define: $\vec{k}(s, t) = -\frac{\gamma}{2\pi} \int_s^t \vec{G}(\tau) d\tau$



$$M_{xy}(t, \vec{r}) = i\gamma M_0 \int_0^t B_1(s) e^{i2\pi \vec{k}(s, t) \cdot \vec{r}} ds$$

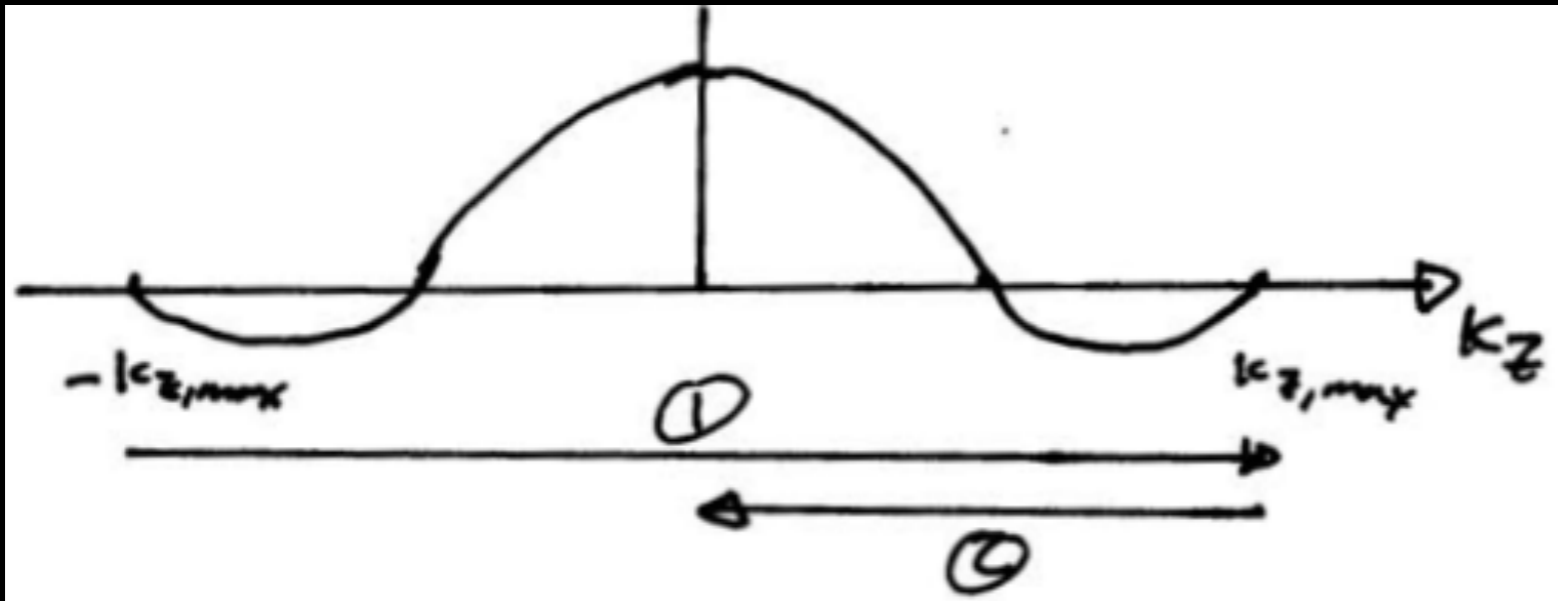
One-Dimensional Example

$$\vec{k}(s, t) = -\frac{\gamma}{2\pi} \int_s^t \vec{G}(\tau) d\tau$$



Consider the value of k at $s = t_1, t_2, \dots, t_7$

One-Dimensional Example



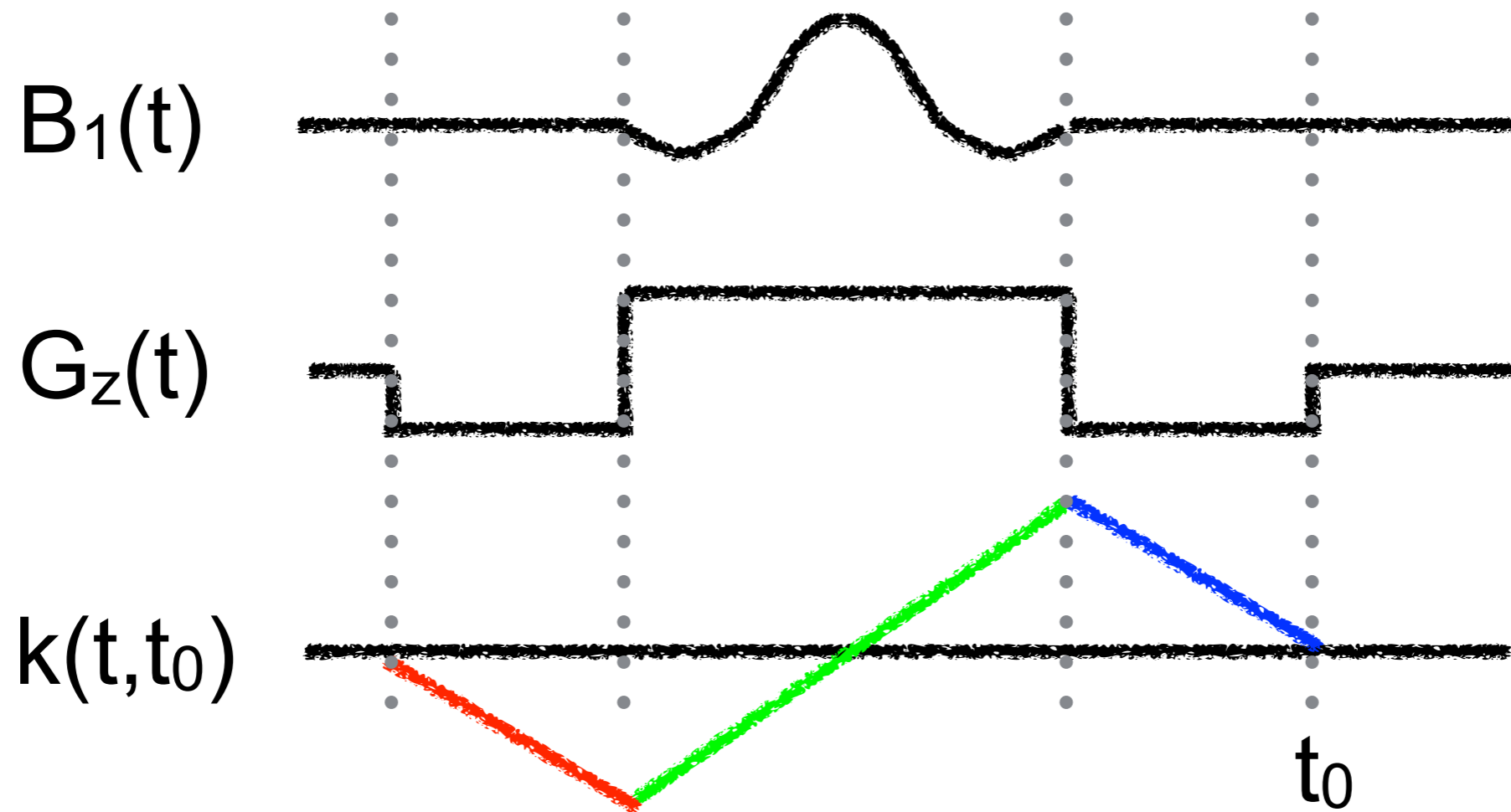
$$k_{z,max} = \frac{T}{2} \frac{\gamma}{2\pi} G_z$$

- This gives magnetization at $t = t_0$, the end of the pulse
- Looks like you scan across k-space, then return to origin

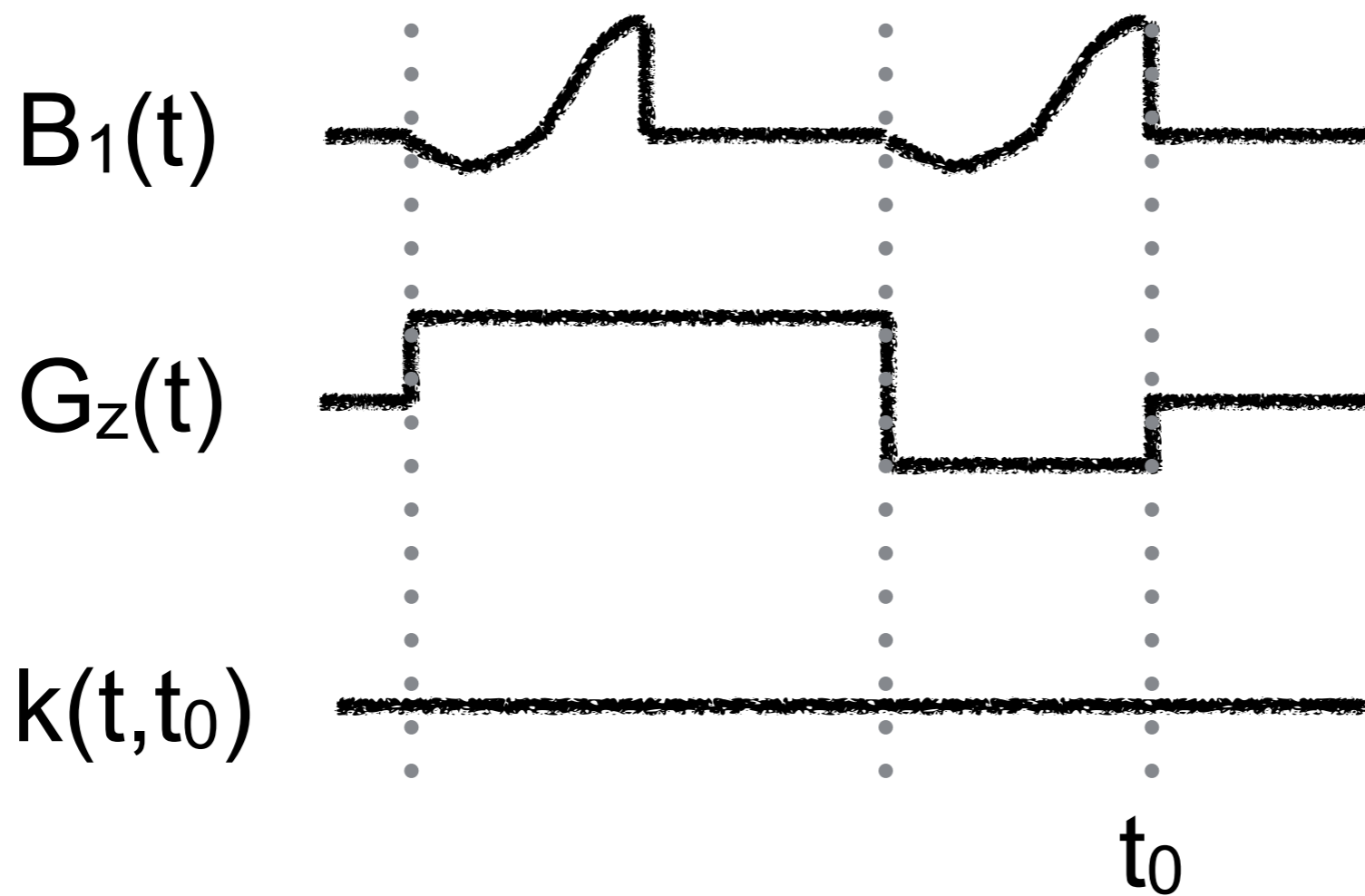
Evolution of Magnetization During Pulse

- RF pulse goes in at DC ($k_z = 0$)
- Gradients move previously applied weighting around
- Think of the RF as “writing” an analog waveform in k-space
- Same idea applies to reception

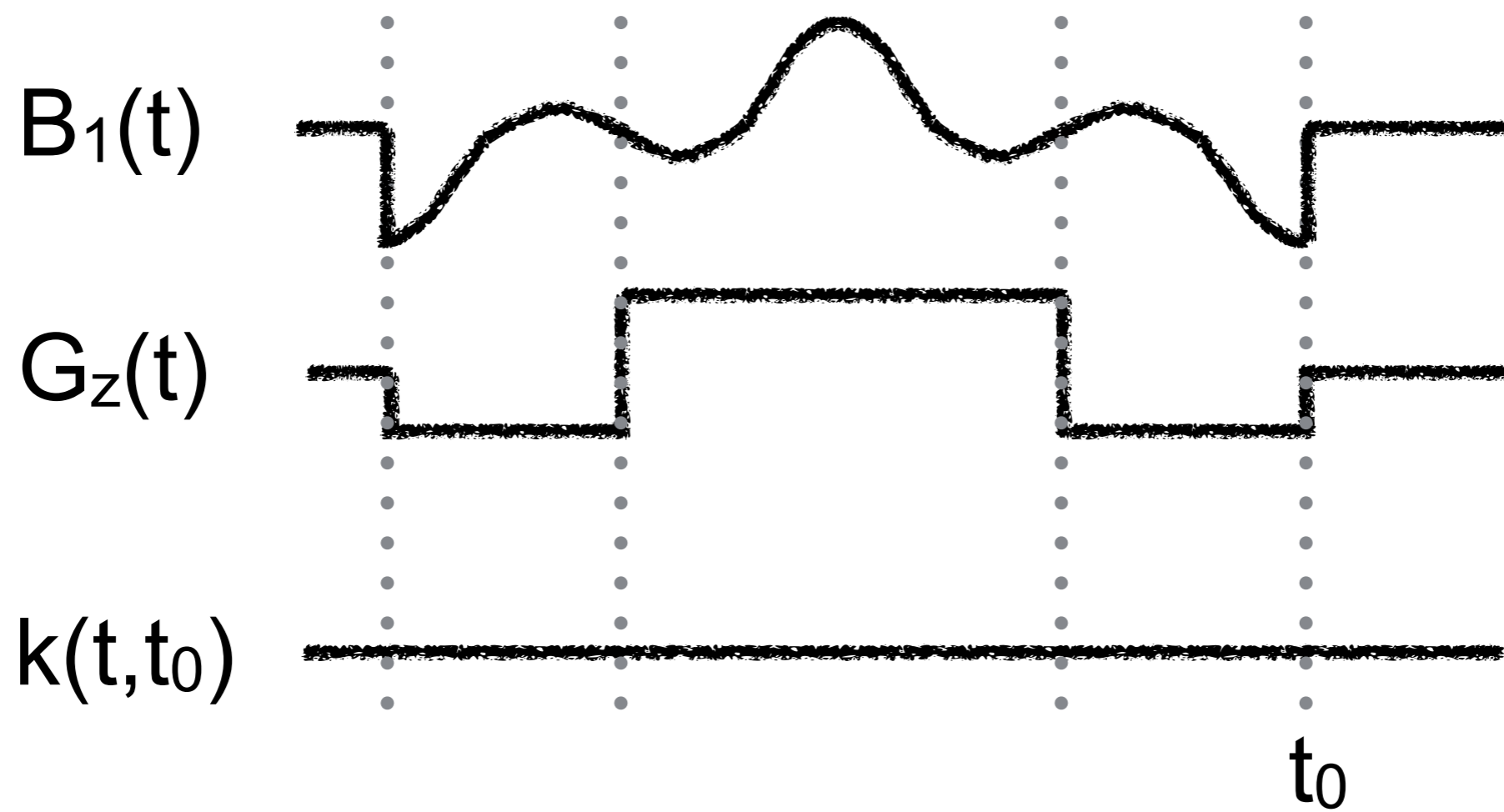
Other 1D Examples



Other 1D Examples



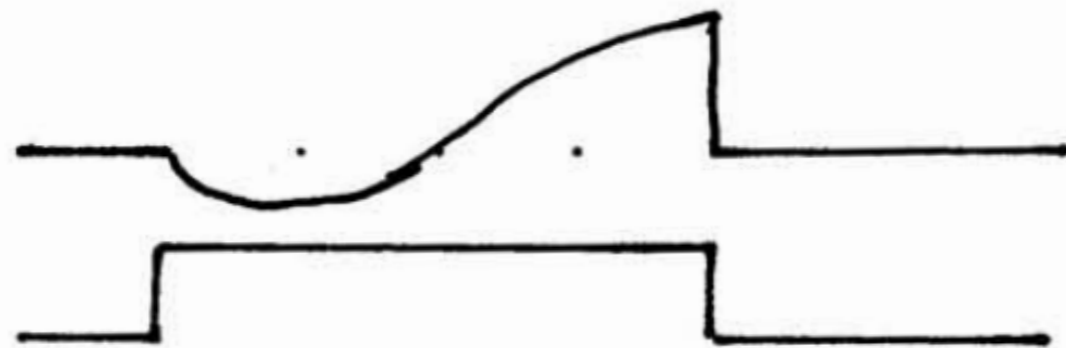
Other 1D Examples



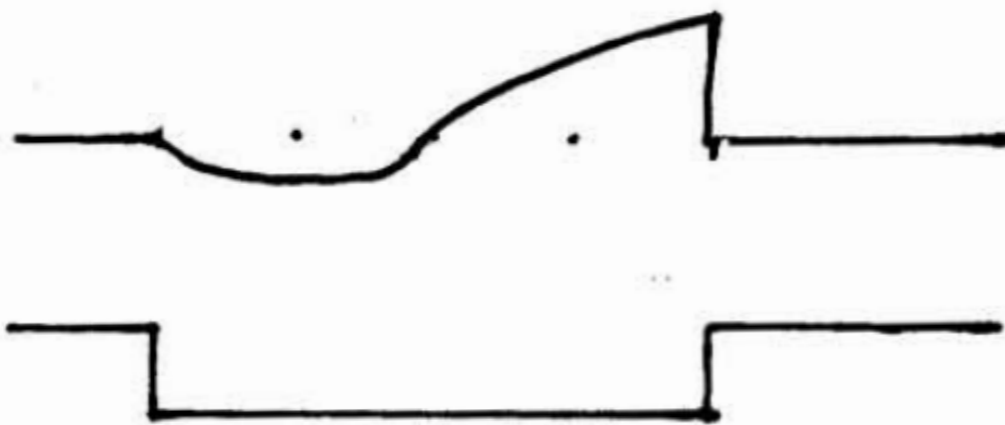
Multiple Excitations

- Most acquisition methods require several repetitions to make an image
 - e.g., 128 phase encodes
- Data is combined to reconstruct an image
- Same idea works for excitation!

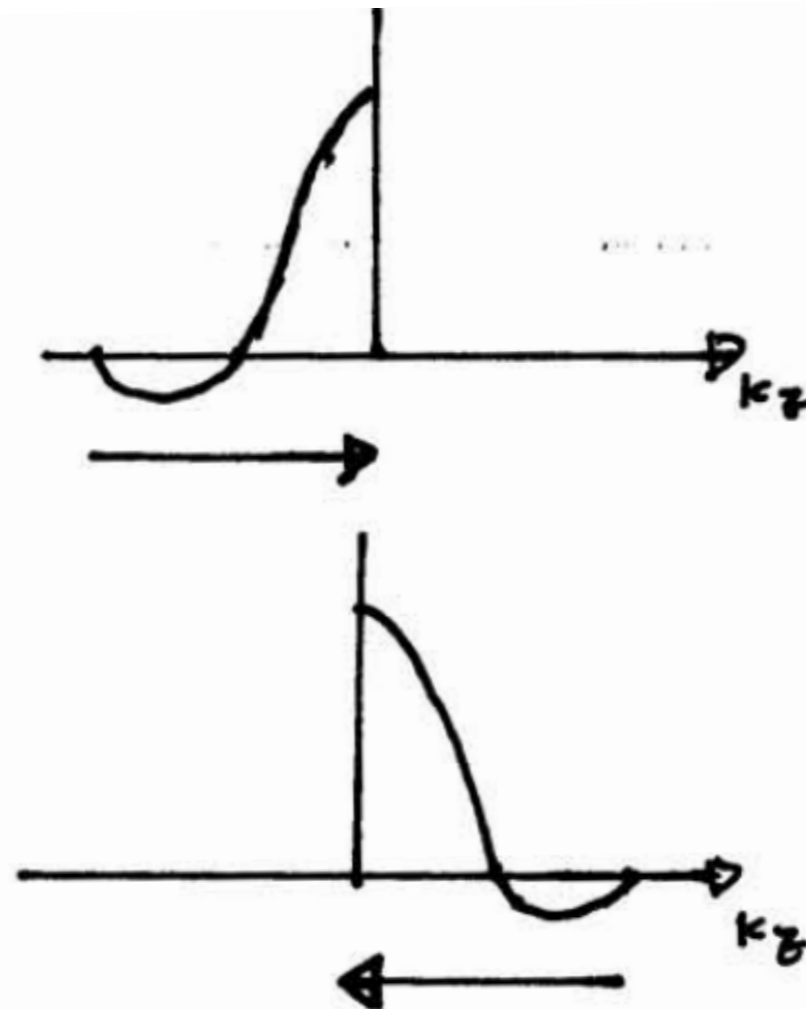
Simple 1D Example



FIRST REPEATITION



SECOND REPEATITION



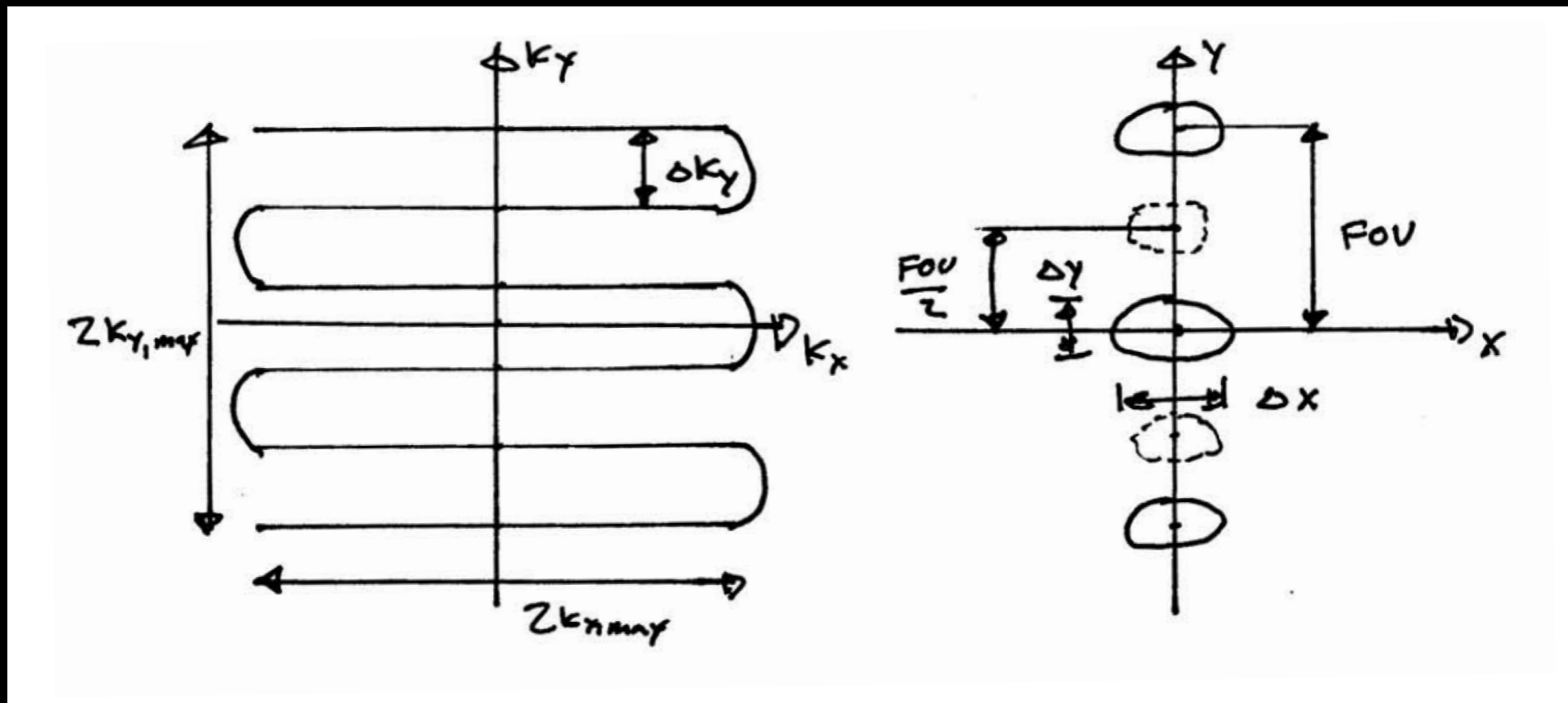
Sum the data from two acquisitions

Same profile as slice selective pulse, but zero echo time

2D EPI Pulse Design

Designing EPI k-space Trajectory

- Ideally, an EPI trajectory scans a 2D raster in k-space



Resolution? / FOV?

Designing EPI k-space Trajectory

- Resolution: $\Delta x = \frac{TBW}{2k_{x,max}}$ $\Delta y = \frac{TBW}{2k_{y,max}}$

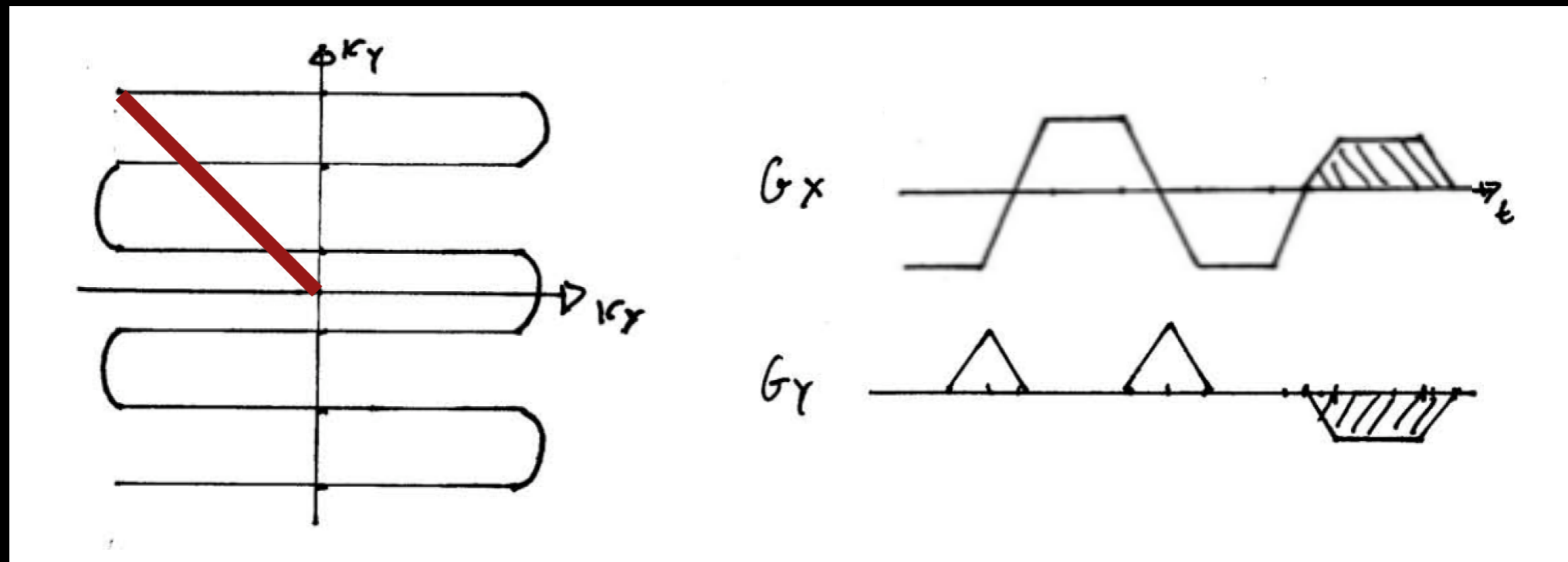
- FOV = $1/\Delta k_y$ $\Delta k_y = \frac{2k_{y,max}}{L-1}$

- Ghost FOV = FOV/2

- Eddy currents & delays produce this

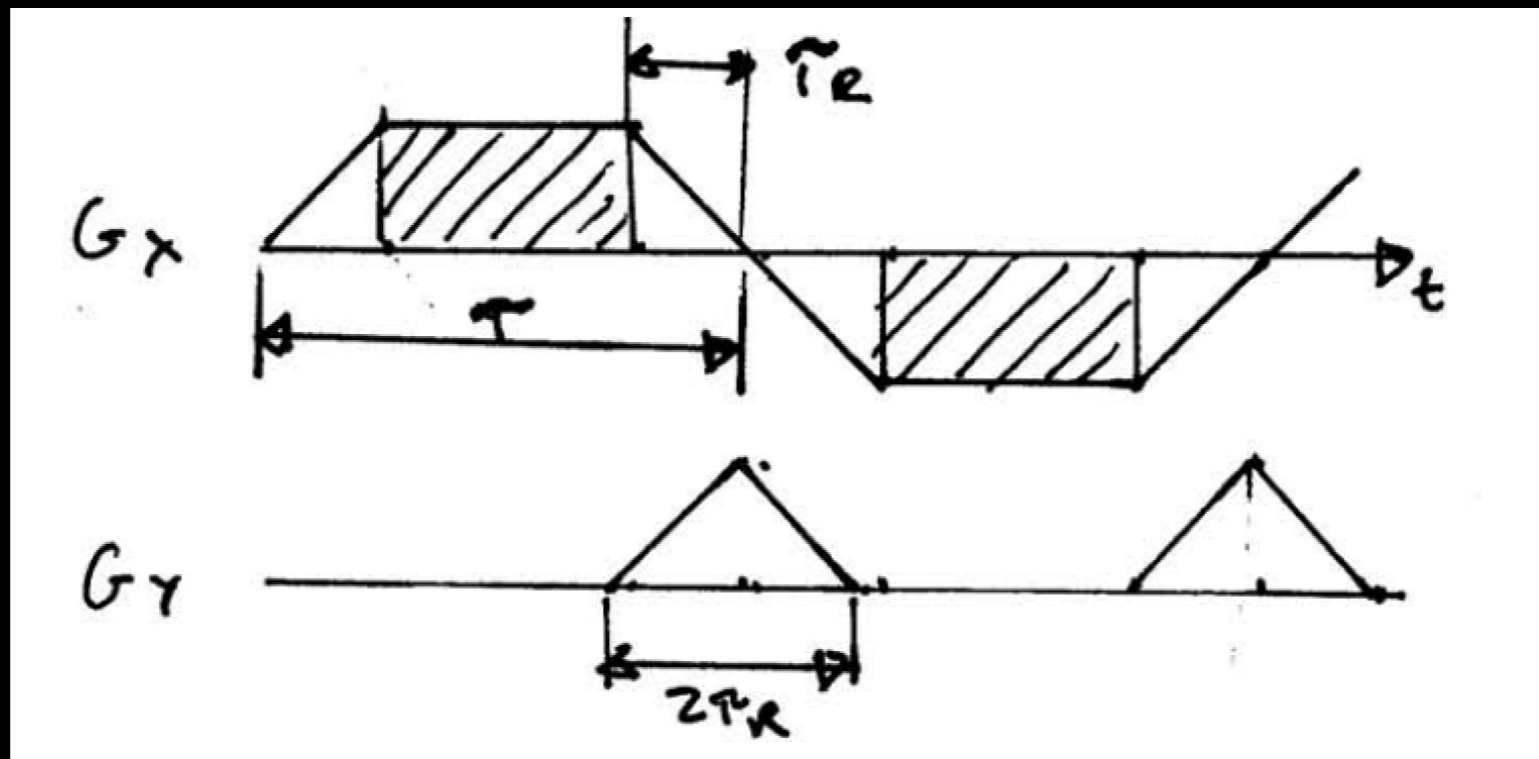
Designing EPI k-space Trajectory

- Refocusing gradients
 - Returns to origin at the end of pulse



Designing EPI Gradients

- Designing readout lobes and blips
 - Flat-top only design



- RF only played during flat part (simpler)

To the board ...

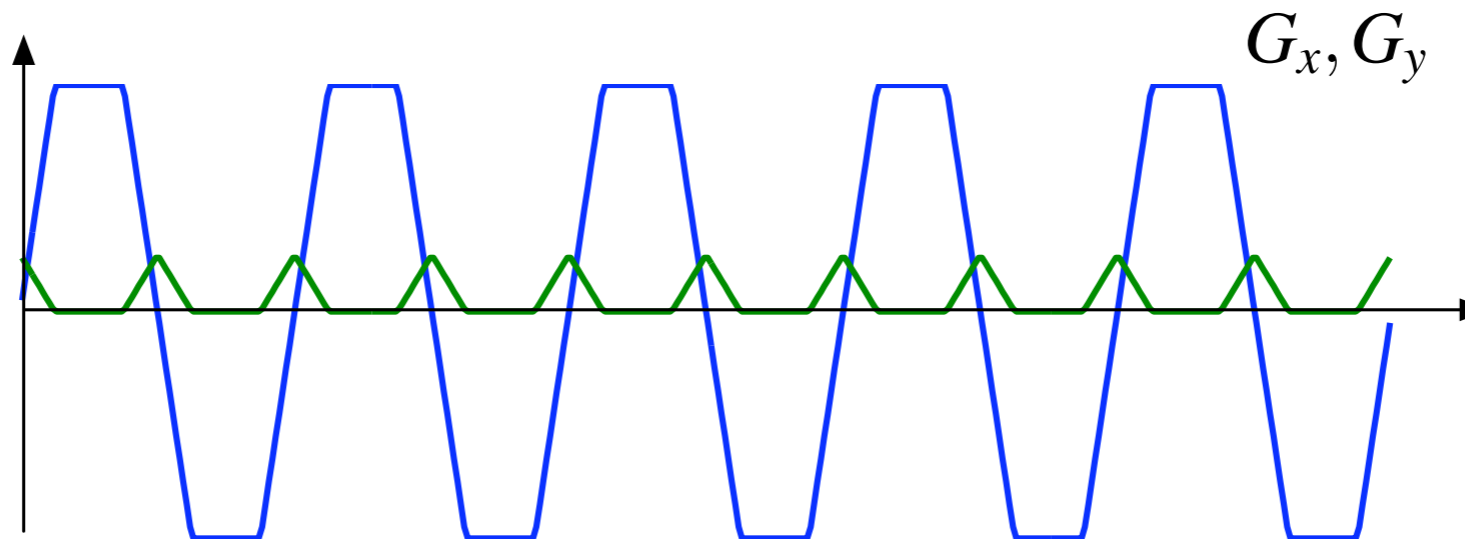
Designing EPI Gradients

- Easy to get k-space coverage in k_y
- Hard to get k-space coverage in k_x
- We can get more k-space coverage by
 - making blips narrower
 - playing RF during part of ramps

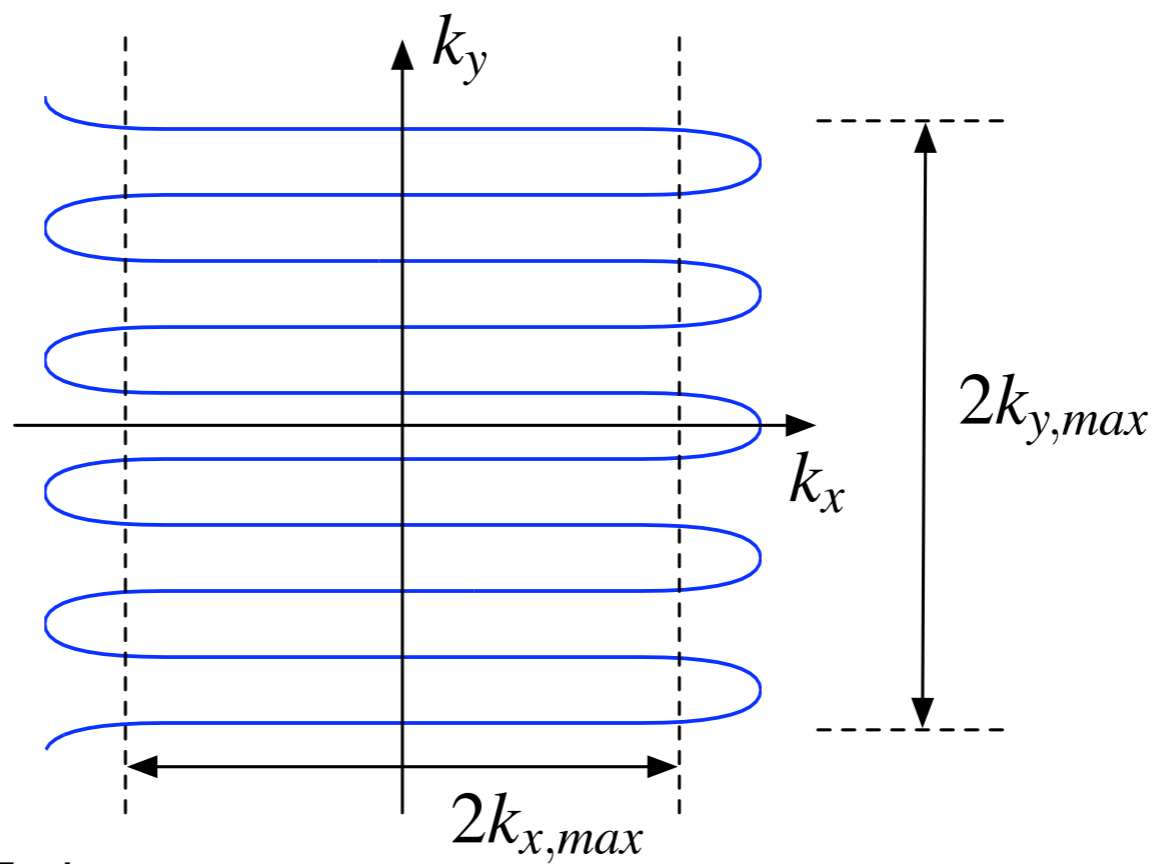
Blipped EPI

- Rectilinear scan of k-space
- Most efficient EPI trajectory
- Common choice for spatial pulses
- Sensitive to eddy currents and gradient delays

Blipped EPI



Gradient Waveforms

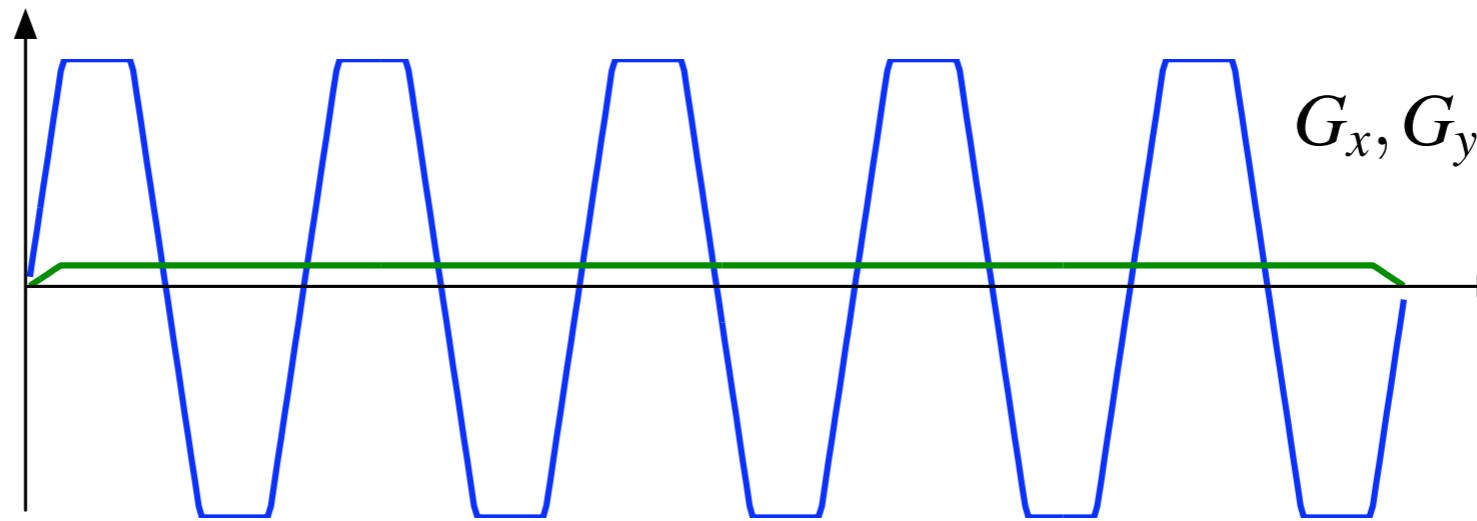


k-Space Trajectory

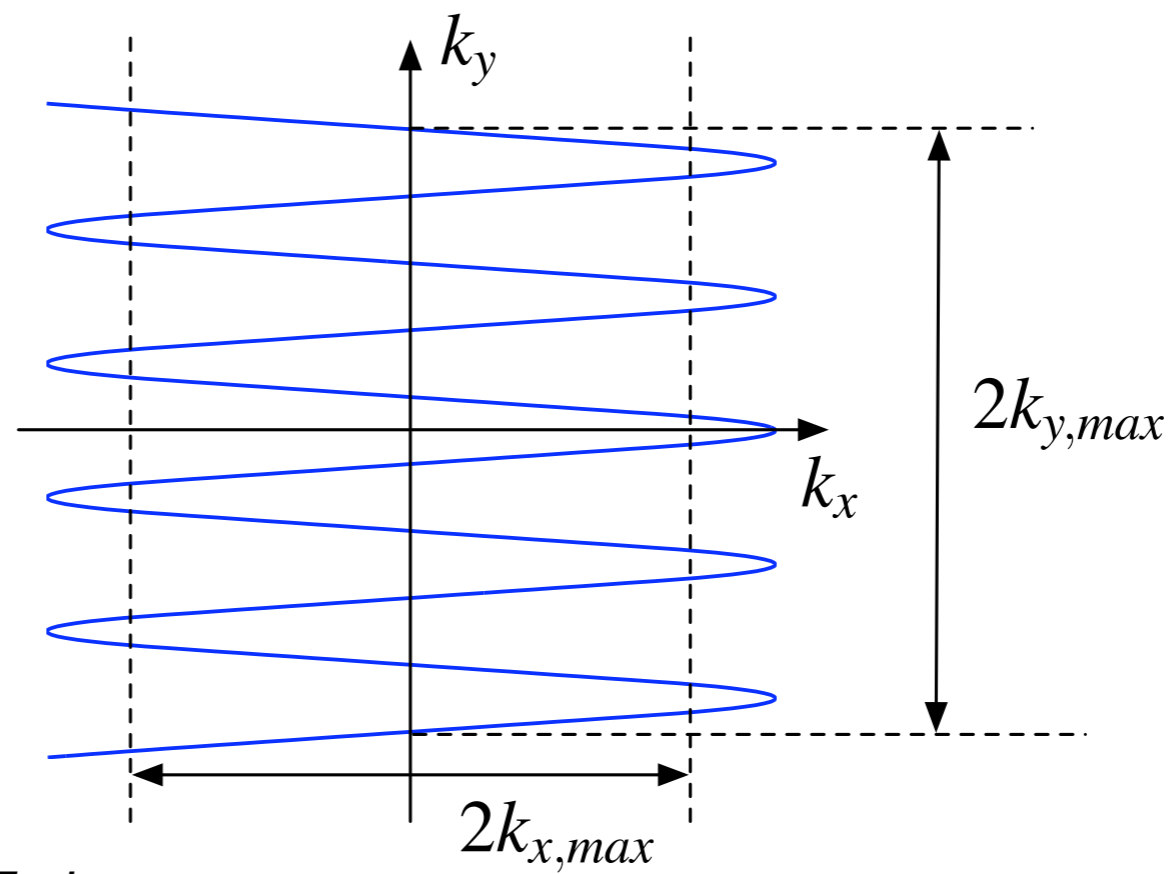
Continuous EPI

- Non-uniform k-space coverage
- Need to oversample to avoid side lobes
 - Less efficient than blipped
- Sensitive to eddy currents and gradient delays
 - Only choice for spectral-spatial pulses

Continuous EPI



Gradient Waveforms

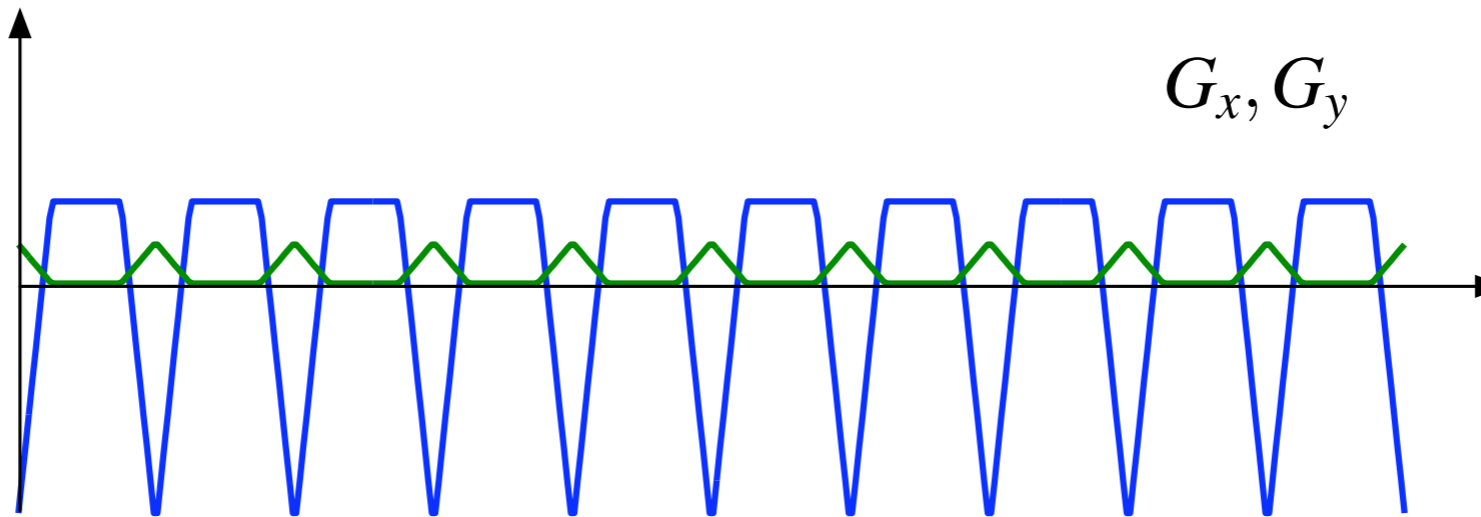


k -Space Trajectory

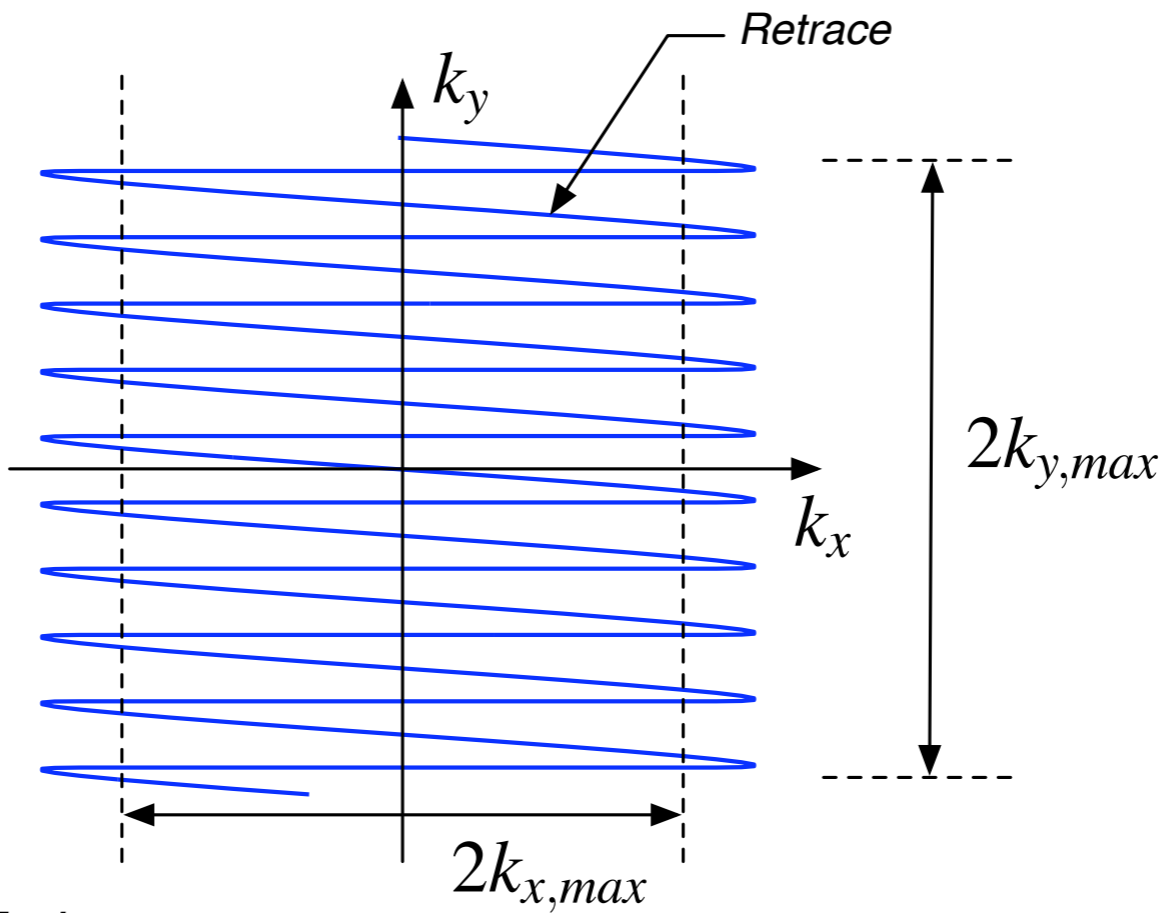
Flyback EPI

- Can be blipped or continuous
- Less efficient since retraces not used (depends on gradient system)
- Almost completely immune to eddy currents and gradient delays

Flyback EPI



Gradient Waveforms



k-Space Trajectory

Designing 2D EPI Spatial Pulses

- Two major options
 - General approach, same as 2D spiral pulses
 - Seperable, product design (easier)
- General approach
 - Choose EPI k-space trajectory
 - Design gradient waveforms
 - Design $W(k)$, k-space weighting
 - Design $B_1(t)$

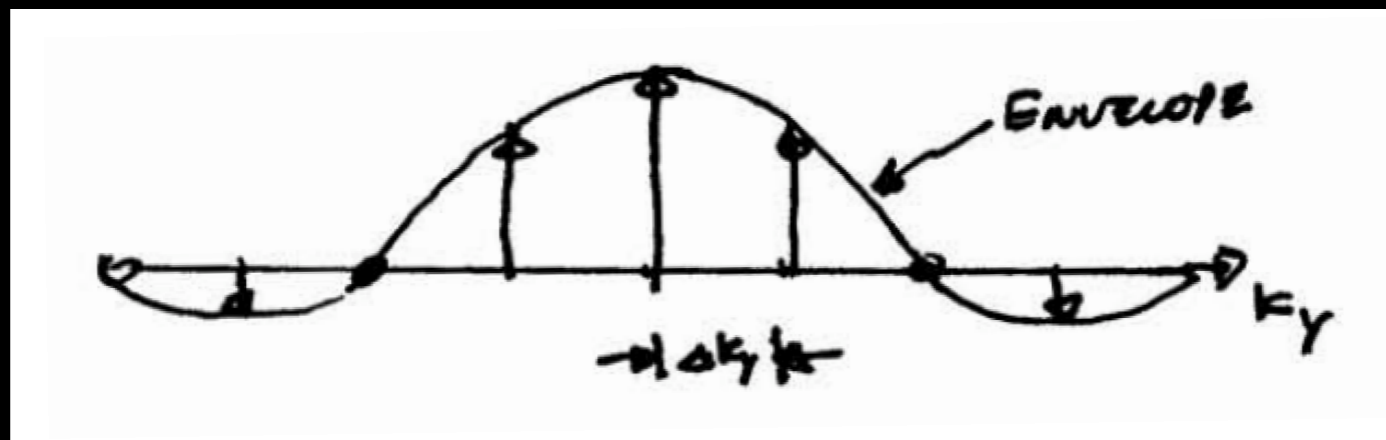
Separable, Product Design

- Assume,

$$W(k_x, k_y) = A_F(k_x) \cdot A_S(k_y)$$

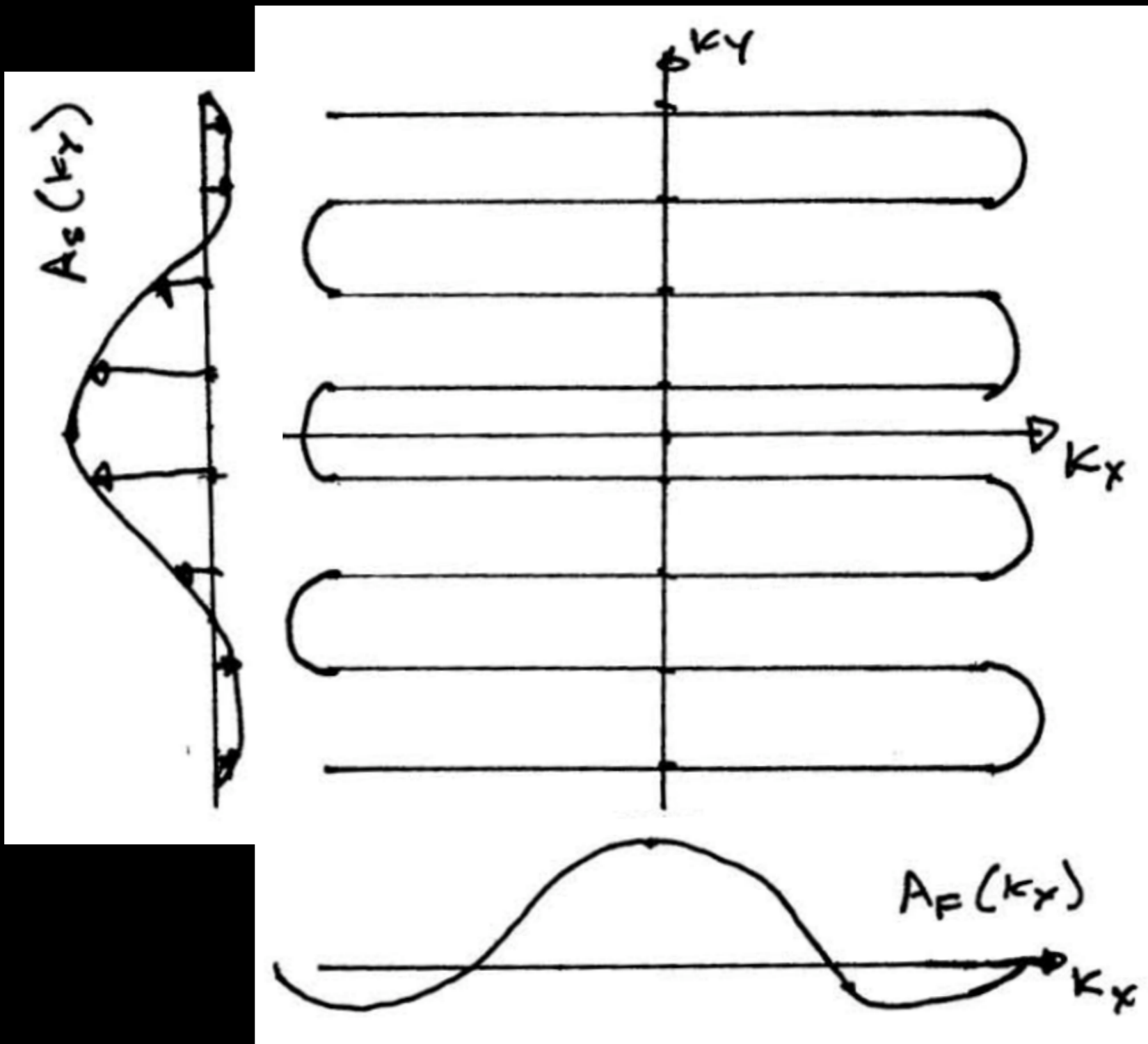
$A_S(k_y)$: weighting in the slow, blipped direction

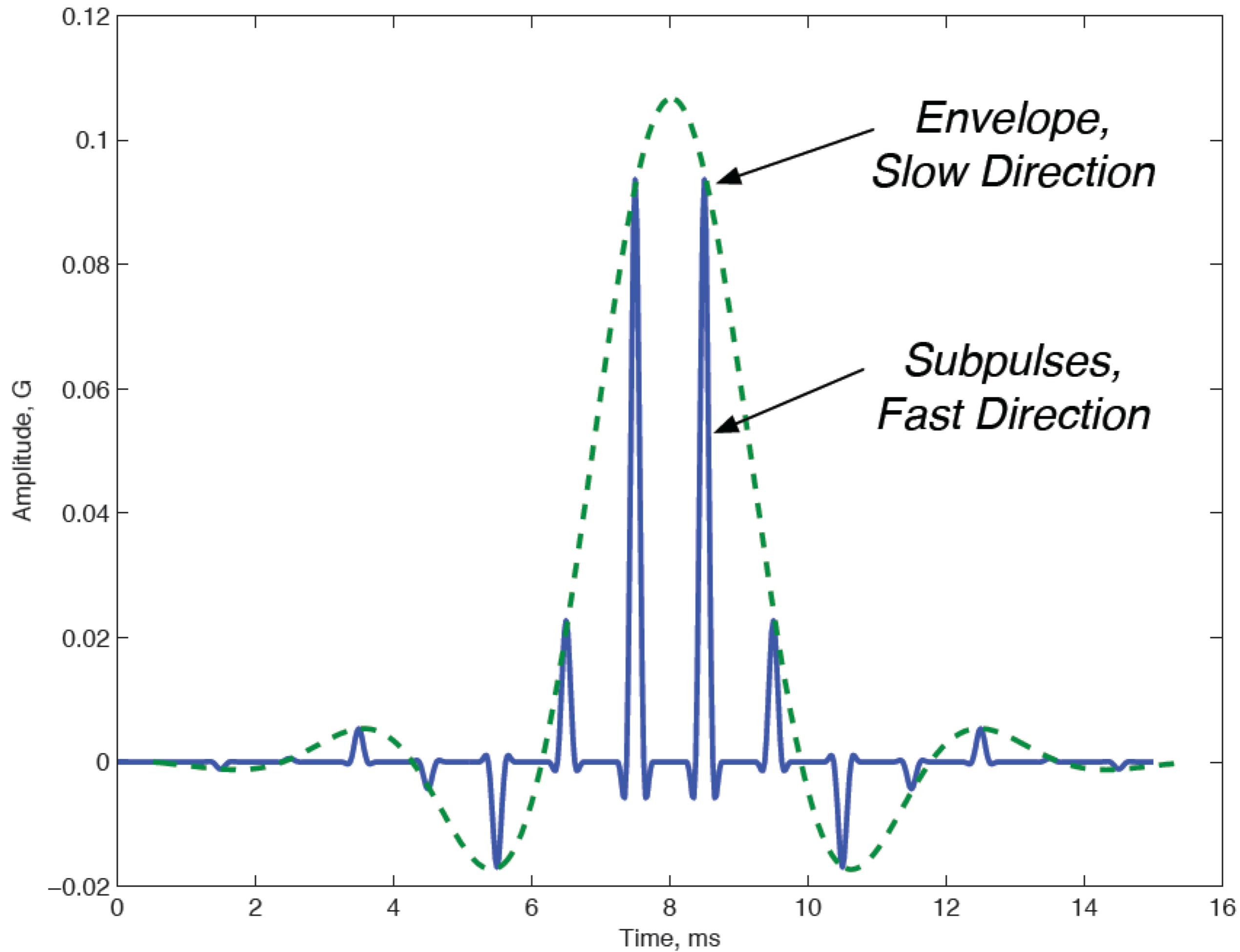
$A_F(k_x)$: weighting in the fast oscillating direction

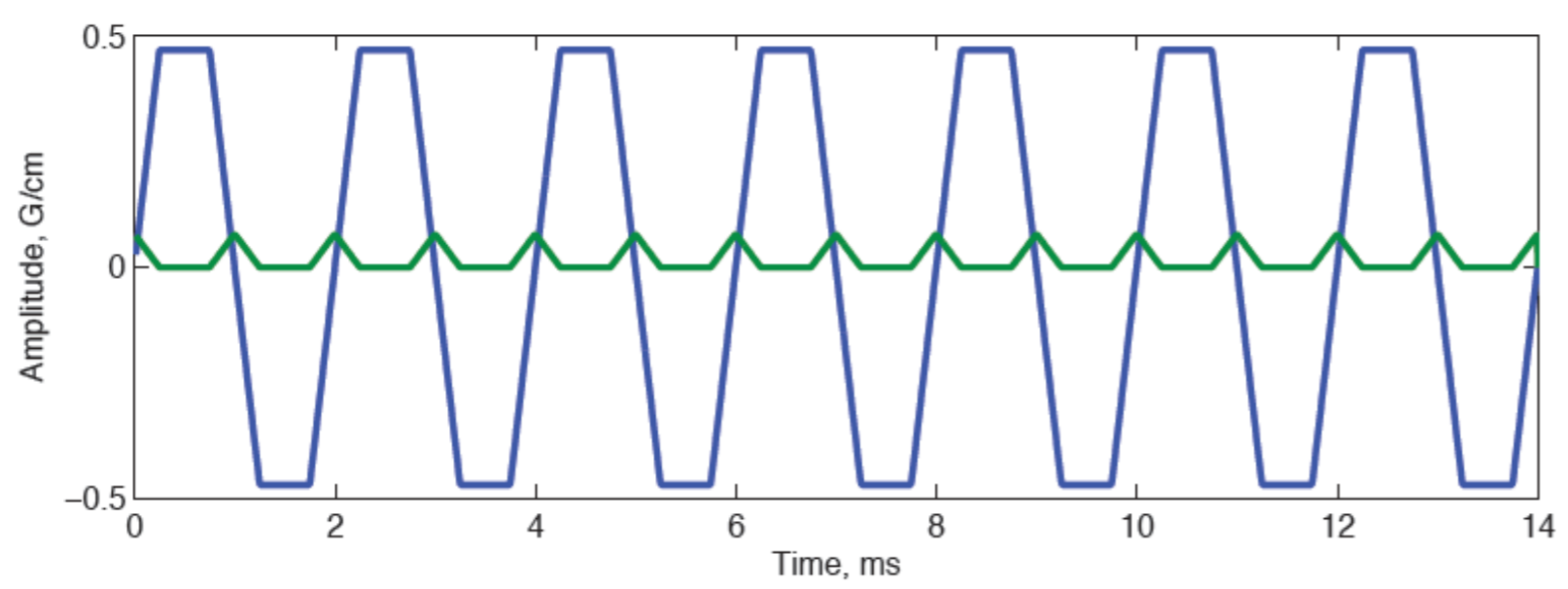
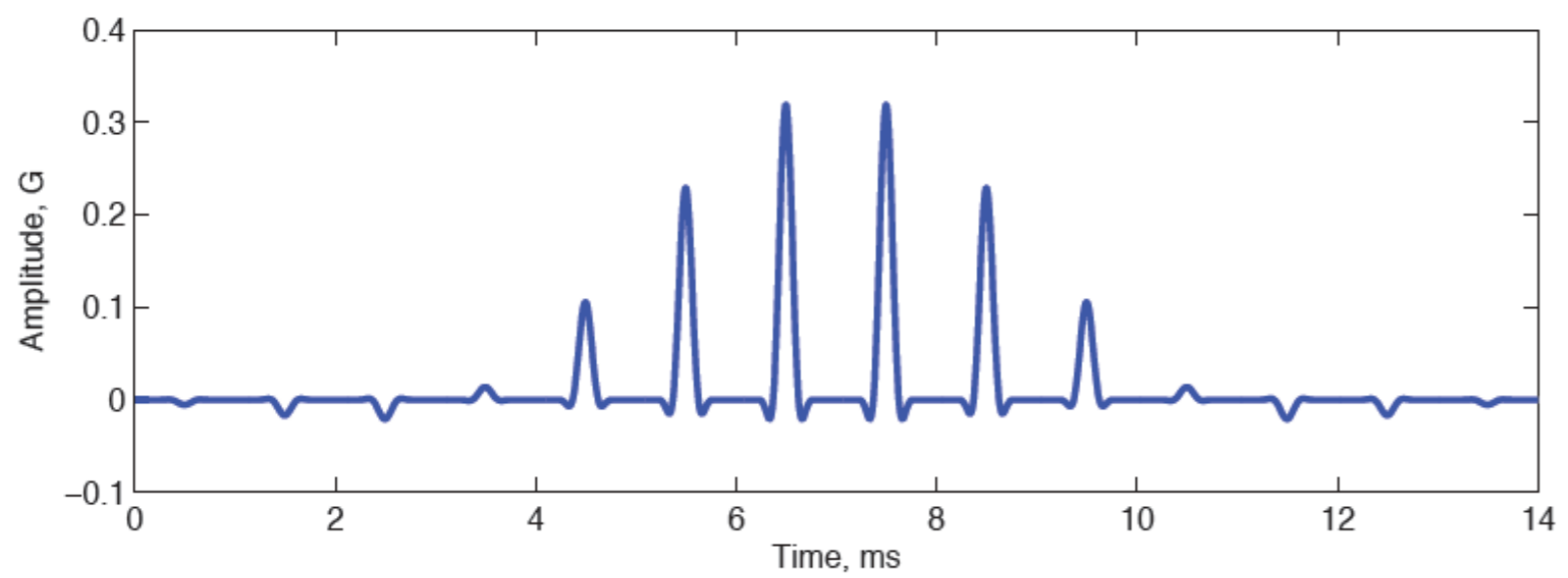


- Each impulse corresponds to a pulse in the fast direction, $A_F(k_x)$

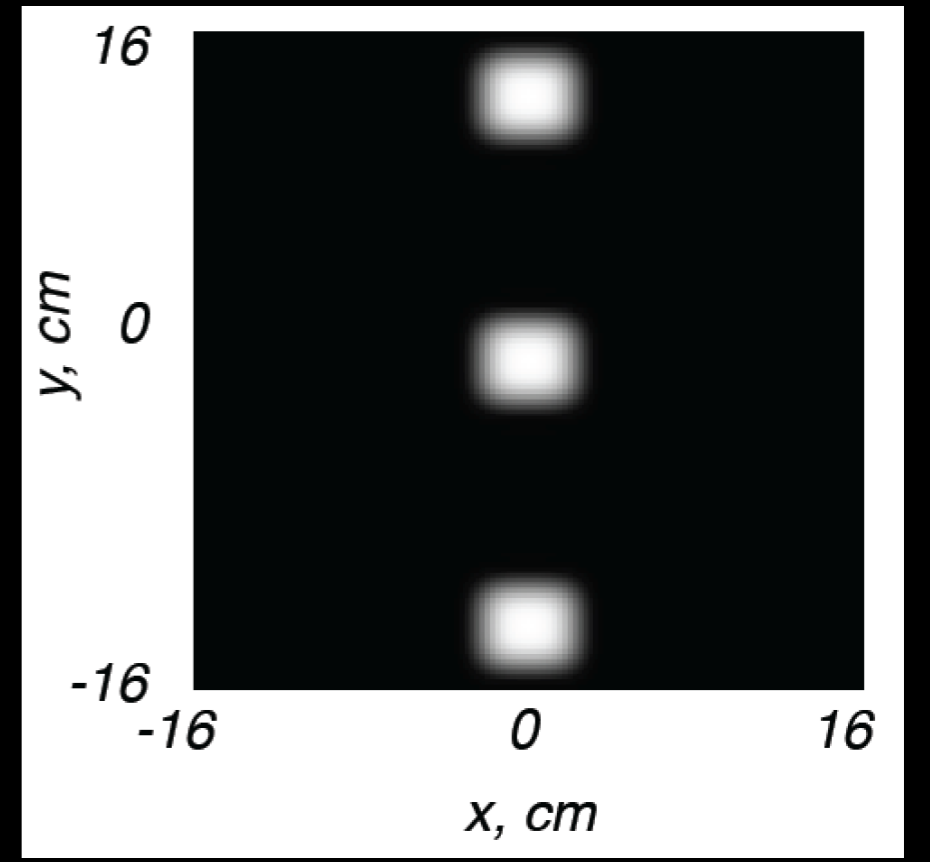
Separable, Product Design







1 ms subpulses
14 subpulses
Flattop only (0.5 ms)
4 cm x 4 cm mainlobe
Sidelobes at +/- 13 cm



Matlab Exercise

Bloch Simulator

- <http://mrsrl.stanford.edu/~brian/blochsim/>

```
[mx,my,mz] = bloch(b1,gr,tp,t1,t2,df,dp,mode,mx,my,mz)
```

Bloch simulation of rotations due to B1, gradient and off-resonance, including relaxation effects. At each time point, the rotation matrix and decay matrix are calculated. Simulation can simulate the steady-state if the sequence is applied repeatedly, or the magnetization starting at m0.

INPUT:

b1 = (Mx1) RF pulse in G. Can be complex.
gr = (Mx1,2,or 3) 1,2 or 3-dimensional gradient in G/cm.
tp = (Mx1) time duration of each b1 and gr point, in seconds,
or 1x1 time step if constant for all points
or monotonically INCREASING endtime of each
interval..
t1 = T1 relaxation time in seconds.
t2 = T2 relaxation time in seconds.
df = (Nx1) Array of off-resonance frequencies (Hz)
dp = (Px1,2,or 3) Array of spatial positions (cm).
Width should match width of gr.
mode= Bitmask mode:
Bit 0: 0-Simulate from start or M0, 1-Steady State
Bit 1: 1-Record m at time points. 0-just end time.

Windowed Sinc RF Pulse

```
%% Design of Windowed Sinc RF Pulses
```

```
tbw = 4;
```

```
samples = 512;
```

```
rf = wsinc(tbw, samples);
```

```
function h = wsinc(tbw, ns)
```

```
% rf = wsinc(tbw, ns)
```

```
%
```

```
%   tbw   --   time bandwidth product
```

```
%   ns    --   number of samples
```

```
%   h     --   windowed sinc function, normalized so that sum(h) = 1
```

```
xm = (ns-1)/2;
```

```
x = [-xm:xm]/xm;
```

```
h = sinc(x*tbw/2) .* (0.54+0.46*cos(pi*x));
```

```
h = h/sum(h);
```

RF Pulse Scaling

```
%% Plot RF Amplitude
```

```
rf = (pi/2)*wsinc(tbw,samples);
```

```
pulseduration = 1; %ms
```

```
rfs = rfscalerf(rf, pulseduration); % Scaled to Gauss
```

$$\theta = \int_0^{\tau} \gamma B_1(s) ds$$

$$\theta_i = \gamma B_1(t_i) \Delta t$$

$$B_1(t_i) = \frac{1}{\gamma \Delta t} \theta_i$$


RF Pulse Scaling

```
%% Plot RF Amplitude
```

```
rf = (pi/2)*wsinc(tbw,samples);
```

```
pulseduration = 1; %ms
```

```
rfs = rfscaleg(rf, pulseduration); % Scaled to Gauss
```

```
function rfs = rfscaleg(rf,t) 
```

```
% rfs = rfscaleg(rf,t)
```

```
%
```

```
% rf -- rf waveform, scaled so sum(rf) = flip angle
```

```
% t -- duration of RF pulse in ms
```

```
% rfs -- rf waveform scaled to Gauss
```

```
%
```

```
gamma = 2*pi*4.257; % kHz*rad/G
```

```
dt = t/length(rf);
```

```
rfs = rf/(gamma*dt);
```

Bloch Simulation

```
%% Simulate Slice Profile
tbw = 4;
samples = 512;

rf = (pi/2)*wsinc(tbw,samples);
pulseduration = 1; %ms

rfs = rfscalg(rf, pulseduration); % Scaled to Gauss
b1 = [rfs zeros(1,samples/2)]; % in Gauss
g = [ones(1,samples) -ones(1,samples/2)]; % in G/cm

x = (-4:.1:4); % in cm
f = (-250:5:250); % in Hz
dt = pulseduration/samples/1e3;
t = (1:length(b1))*dt; % in usec

% Bloch Simulation
[mx,my,mz] = bloch(b1,g,t,1,.2,f,x,0);
mxy=mx+1i*my;
```

Slice Thickness

- Pulse duration = 1 ms
- TBW = 4
- $G_z = 1 \text{ G/cm}$

$$\Delta z = \frac{BW}{\frac{\gamma}{2\pi} G_z}$$

$$\gamma/2\pi = 4.257 \text{ kHz/G}$$

Thanks!

- Next time:
 - Project Discussion
 - Homework 2
 - 2D EPI design
 - SPSP design

Kyung Sung, PhD

ksung@mednet.ucla.edu

<http://kyungs.bol.ucla.edu>