Vendor-neutral pulse sequence prototyping with Pulseq

Univ. Prof. Dr. Maxim Zaitsev

High Field Magnetic Resonance Center
Center for Medical Physics and Biomedical Engineering
Medical University of Vienna

Pulseq

http://pulseq.github.io

Speaker name:
Prof. Maxim Zaitsev, Medical University of Vienna, Vienna, Austria, Earth

Conflicts of interest regarding this presentation:
Nothing to disclose

ESMRMB
European Society for Magnetic Resonance in Medicine and Biology

Overview

- *Pulseq* sequence definition and programming language overview
  - Pulseq philosophy
  - Pulseq file format
  - Development environments and other options

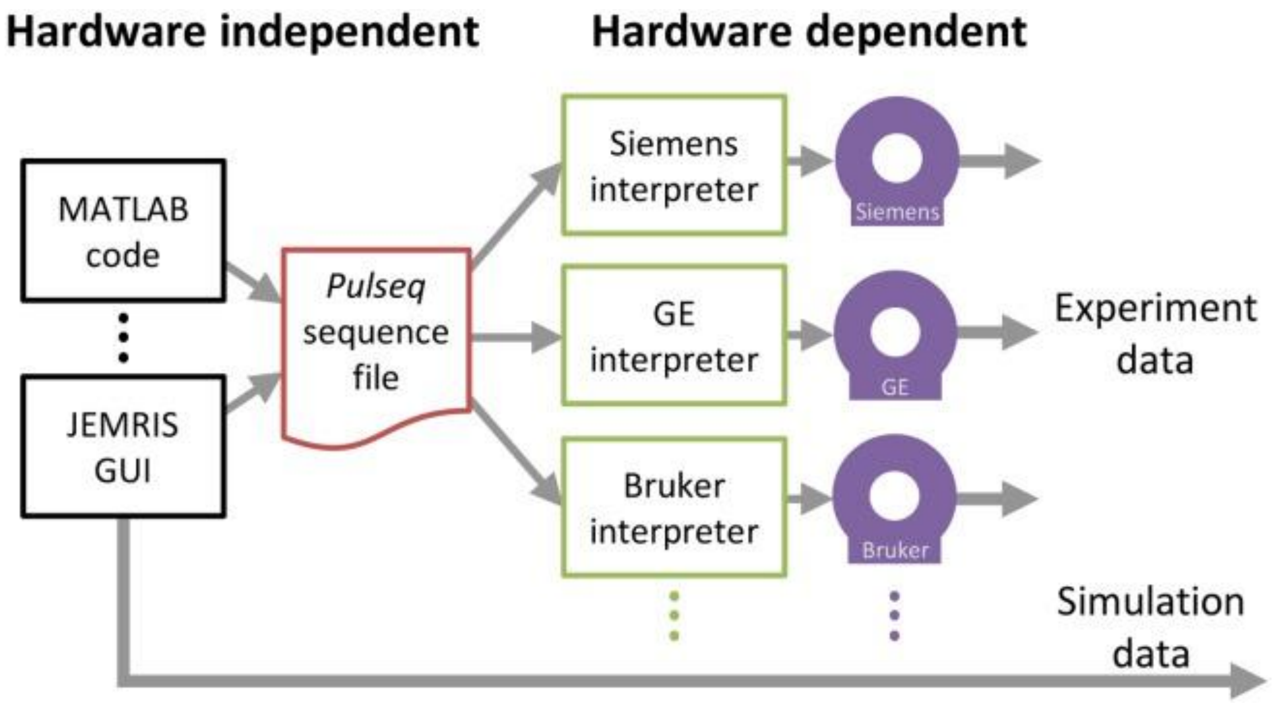
- Using Matlab *Pulseq* Toolbox on Siemens scanners
  - Sequences from scratch: from gradient echo to EPI in 20 minutes

Pulseq Goals

- Remove the initial threshold in sequence programming
  - Make simple things really simple
- Make researcher-oriented features accessible
  - Arbitrary gradients, arbitrary RF, flexible reordering, X-Nuclei, …
- Prevent typical sources of (human) errors
  - Avoid timing errors with overlapping gradients
  - Make flag and counter setting optional/unnecessary
- Minimize effort for implementation and support on hardware
  - Lean sequence-to-hardware interface

Pulseq sequence programming

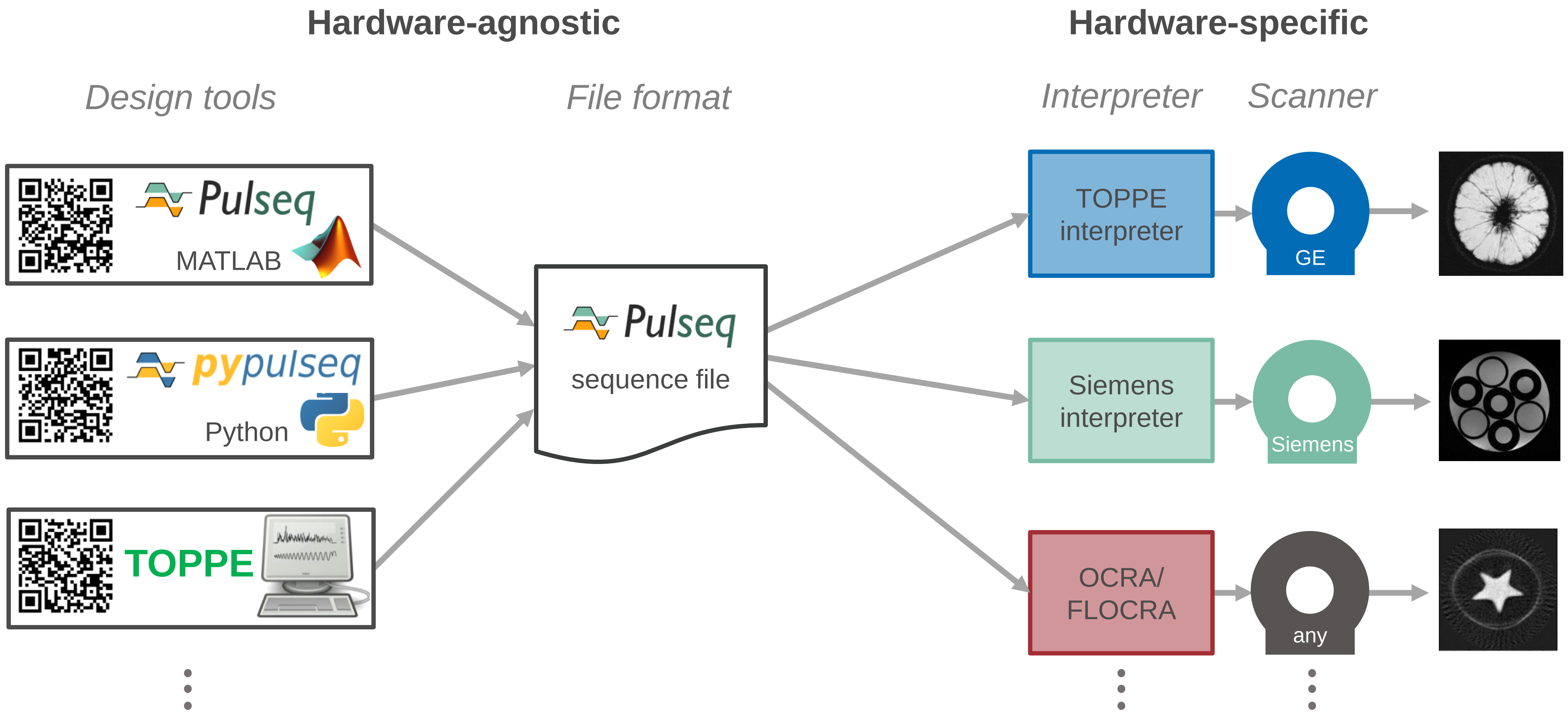
- Cross-platform MRI pulse programming framework
- Low level: *Pulseq* file
- High level: MATLAB[1] or Python[2] toolboxes
- Main goal: ease typical research tasks



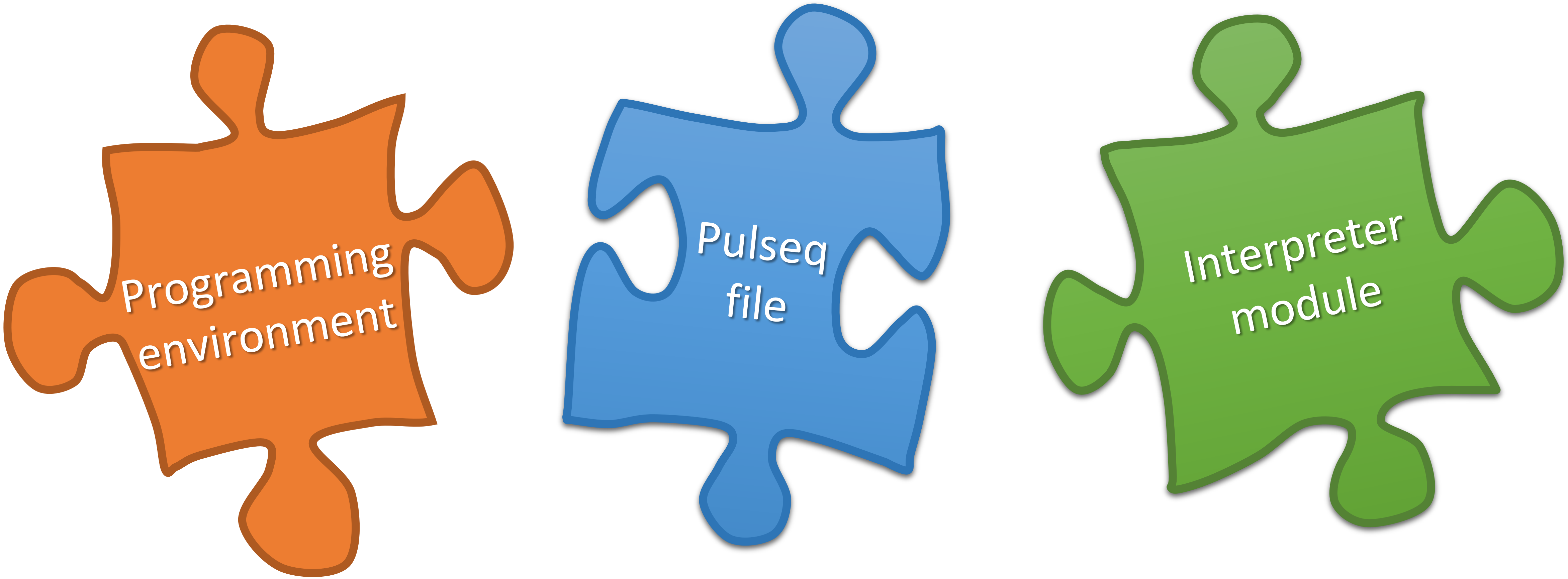
Source: Layton, et al., "Pulseq: A rapid and hardware-independent pulse sequence prototyping framework", MRM 2017

1. http://pulseq.github.io
2. http://github.com/imr-framework/pypulseq

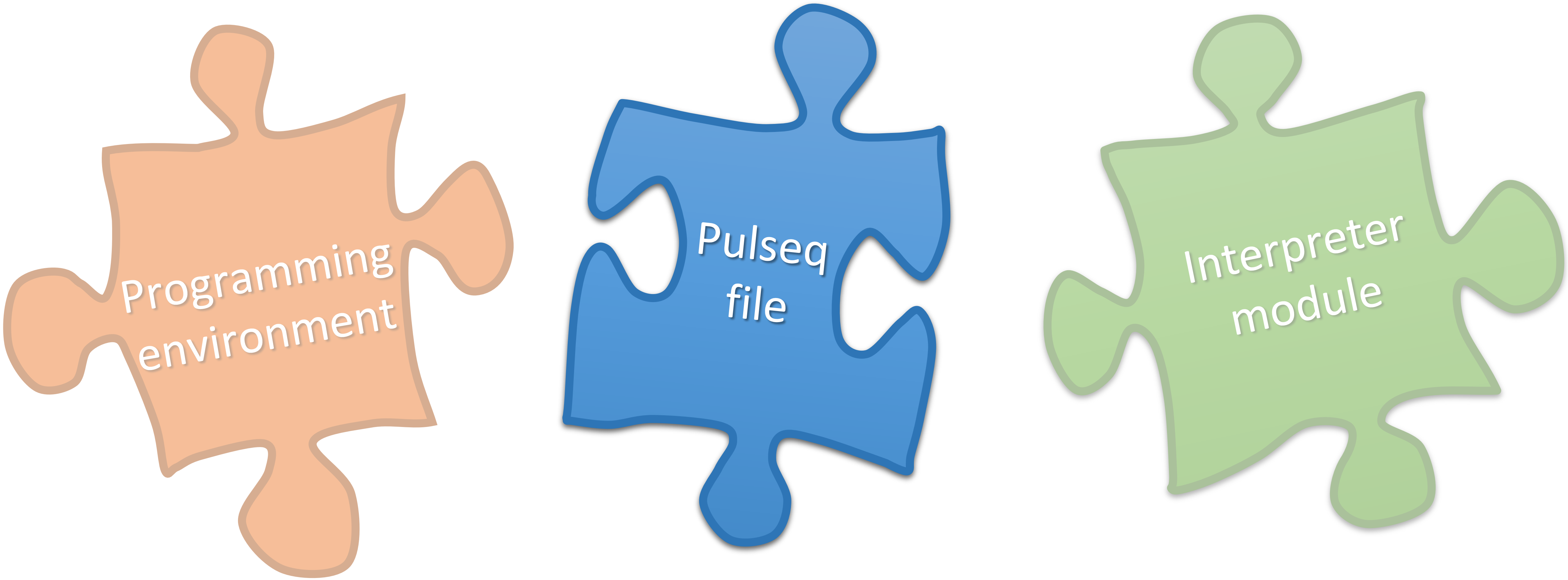
Pulseq framework overview



Pulseq : pieces of the puzzle

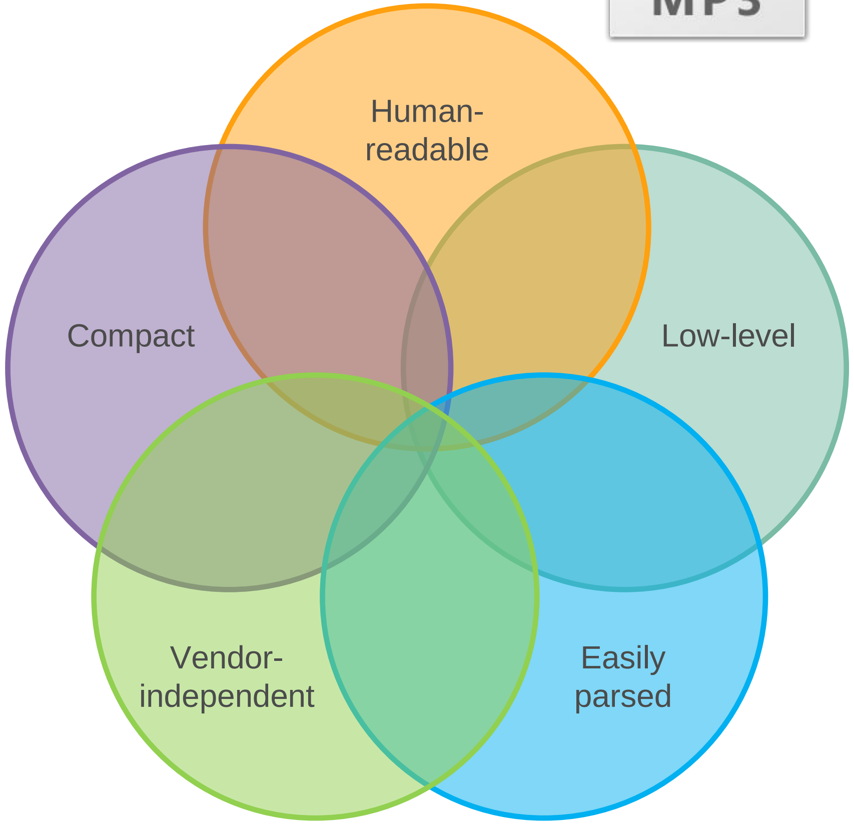


Pulseq : pieces of the puzzle



Pulseq file

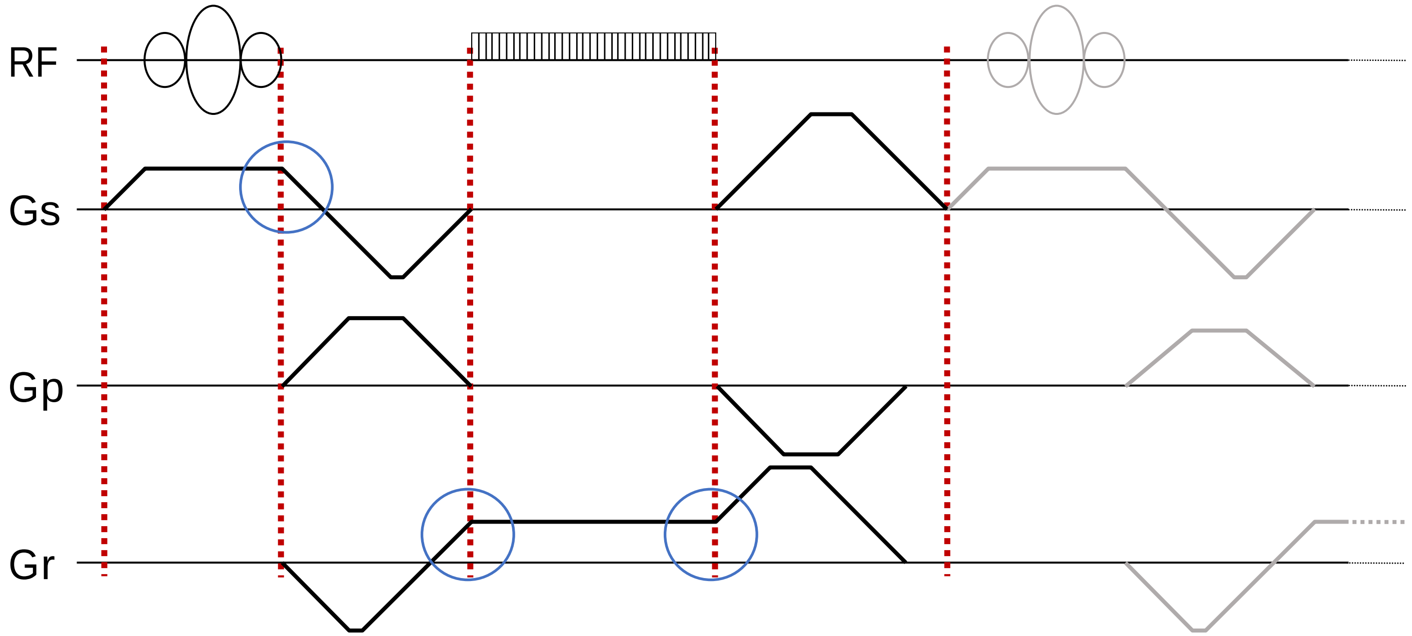
- Explicit (low level) specification of the pulse sequence
  - Think of an MP3 file (or more precisely lossless FLAC)
- No loops, no parameters, no dependencies, no fuss!
- Text file (human-readable)
  - Simple hierarchy
    (RF pulses, gradients, shapes)
  - Event table keeps it together
  - See http://pulseq.github.io/specification.pdf
    for more details



Pulse sequence definition

Pulseq

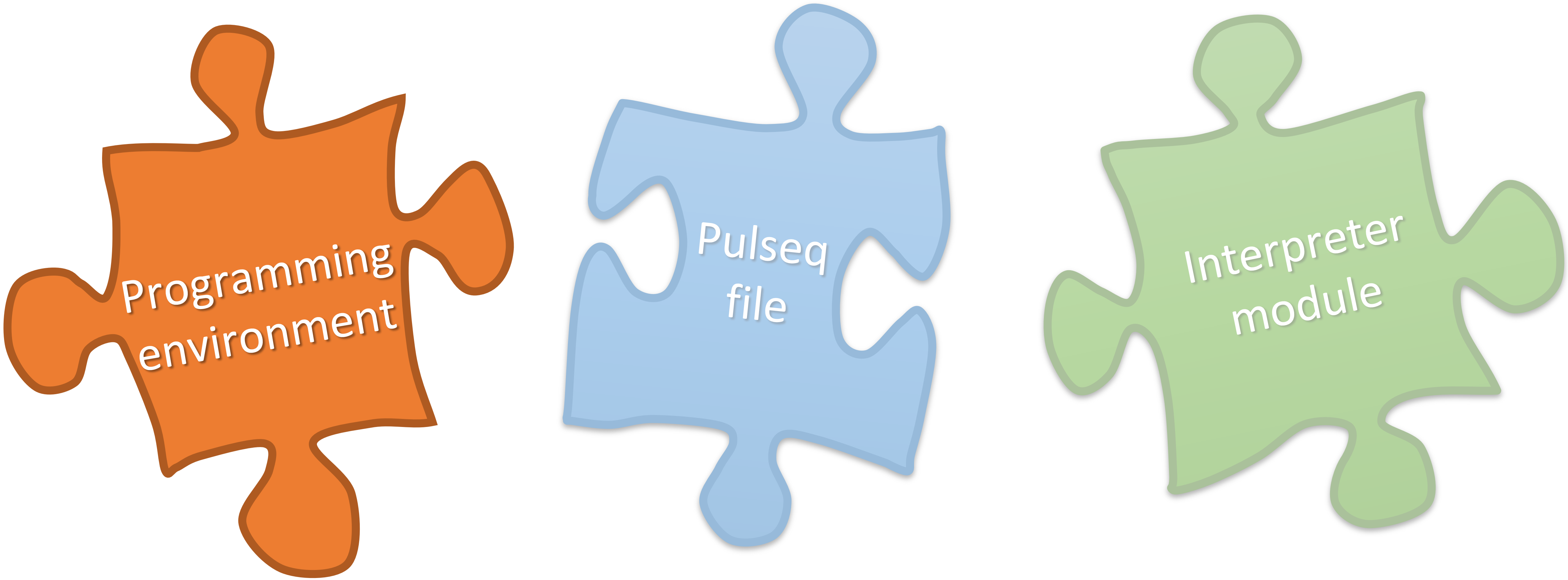
Concatenation of non-overlapping blocks



- Block 1:
  gradient and RF
- Block 2:
  only gradients
- Block 3:
  gradient and ADC
- Block 4:
  only gradients
- Block 5:
  gradient and RF …

Gradients do not have to start or end at 0 at the block boundaries

Pulseq : pieces of the puzzle



High-level programming environments

- Matlab *Pulseq* toolbox
- Python *pypulseq* toolbox
  (see presentation by K. S, Ravi in Session C1 in Caribbean time zone today)

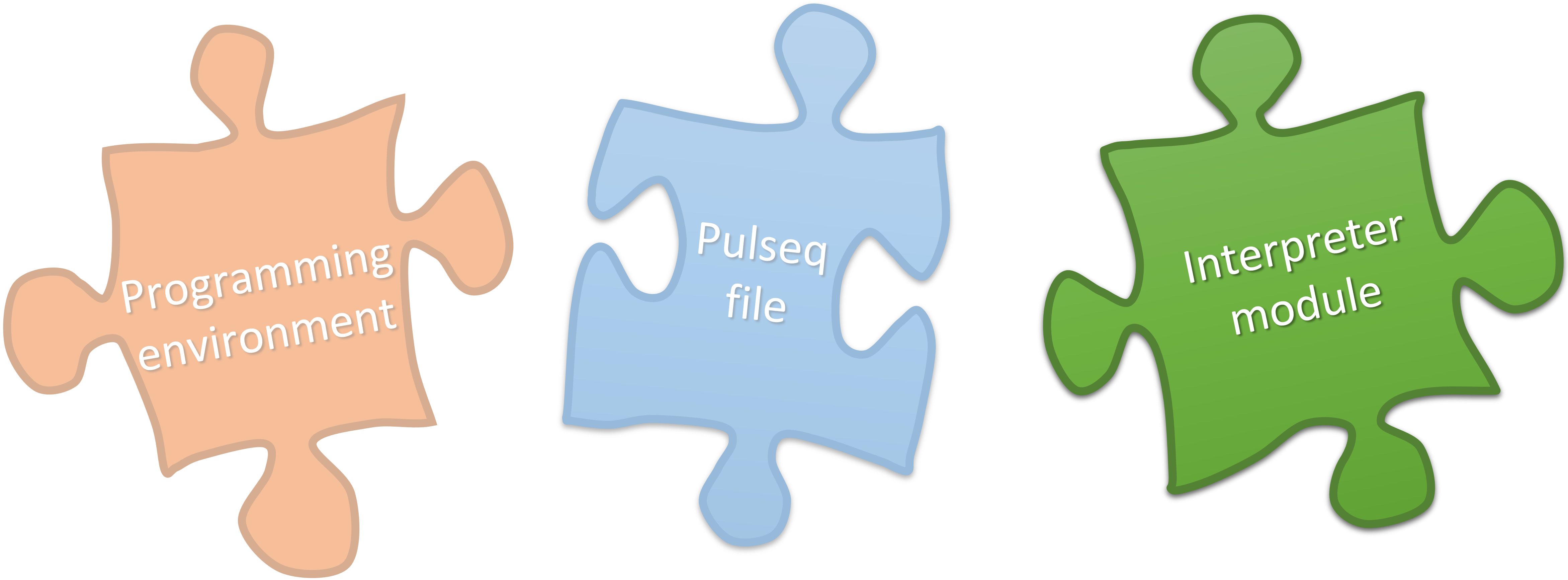
- Further options
  - TOPPE is primarily targeted at GE but can import and export *pulseq* files
    (see hands-on by J.-F. Nielsen in Session C1 in Caribbean time zone today)
  - GammaStar can export *pulseq* files
  - JEMRIS Bloch simulator can export *pulseq* files
  - CoreMRI Bloch simulator can export *pulseq* files
  - …

Matlab Pulseq workflow

- Define the system properties
- Define high-level parameters (convenience)
- Define pulses used in the sequence
- Calculate the delays and reordering tables
- Loop and define sequence blocks
  - Duration of each block is defined by the duration of the longest event
- Copy ‘gre.seq’ to the scanner and run it!
- *Screenshot shows an entire runnable gradient echo sequence code (similar to Siemens’ example miniFlash)*

```matlab
system = mr.opts('MaxGrad',30,'GradUnit','mT/m',...
    'MaxSlew',170,'SlewUnit','T/m/s');
seq=mr.Sequence(system);

fov = 220e-3; Nx=64; Ny=64; TE = 10e-3; TR = 20e-3;

[rf, gz] = mr.makeSincPulse(15*pi/180,system,'Duration',4e-3,...
    'SliceThickness',5e-3,'apodization',0.5,'timeBwProduct',4);

gx = mr.makeTrapezoid('x',system,'FlatArea',Nx/fov,'FlatTime',6.4e-3);
adc = mr.makeAdc(Nx,'Duration',gx.flatTime,'Delay',gx.riseTime);
gxPre = mr.makeTrapezoid('x',system,'Area',-gx.area/2,'Duration',2e-3);
gzReph = mr.makeTrapezoid('z',system,'Area',-gz.area/2,'Duration',2e-3);
phaseAreas = ((0:Ny-1)-Ny/2)*1/fov;

delayTE = TE - mr.calcDuration(gxPre) - mr.calcDuration(rf)/2 ...
    - mr.calcDuration(gx)/2;
delayTR = TR - mr.calcDuration(gxPre) - mr.calcDuration(rf) ...
    - mr.calcDuration(gx) - delayTE;
delay1 = mr.makeDelay(delayTE);
delay2 = mr.makeDelay(delayTR);

for i=1:Ny
    seq.addBlock(rf,gz);
    gyPre = mr.makeTrapezoid('y',system,'Area',phaseAreas(i),...
                                 'Duration',2e-3);
    seq.addBlock(gxPre,gyPre,gzReph);
    seq.addBlock(delay1);
    seq.addBlock(gx,adc);
    seq.addBlock(delay2)
end

seq.write('gre.seq')
```

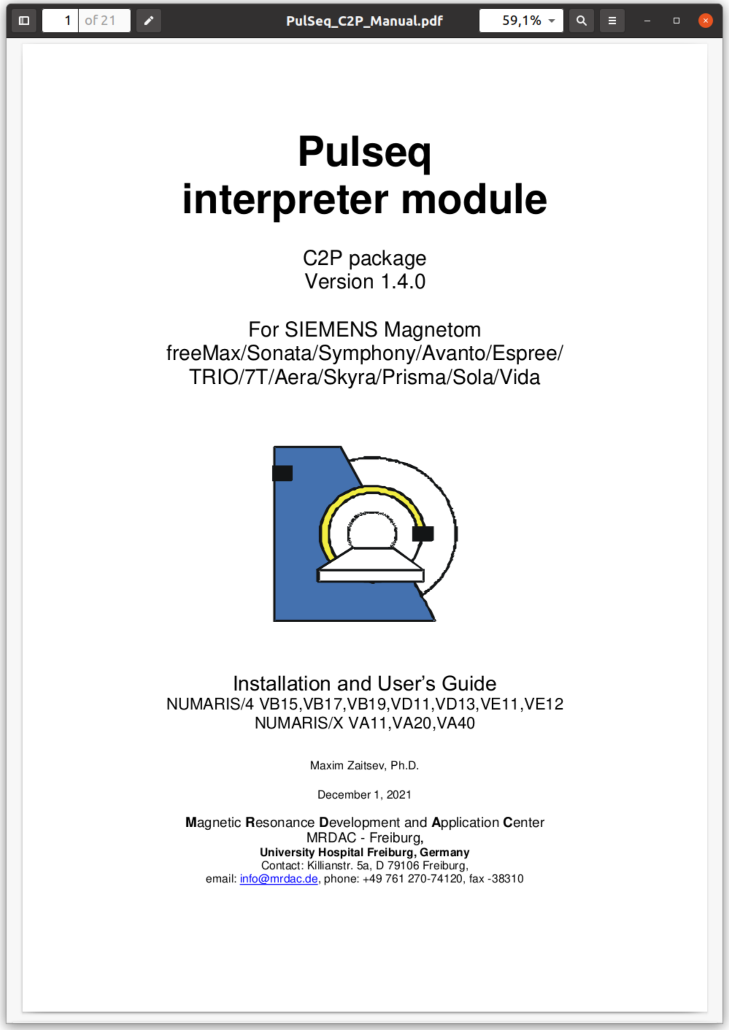
Pulseq : pieces of the puzzle



Pulseq Siemens interpreter

- Just a “normal” sequence
  - Loads its “content” from a Pulseq file
  - All aspects of the sequence are pre-defined
  - FOV positioning and scaling possible
- Based on miniFlash
  - No product code
  - No hacks, no backdoors
- Distributed as a C2P package in source form

- Standard SAR calculation
- Since 1.3.1: libBalance applicable to all sequences
- Safety equal or higher than a typical IDEA sequence


PulSeq_C2P_Manual.pdf

Pulseq
interpreter module

C2P package
Version 1.4.0

For SIEMENS Magnetom
freeMax/Sonata/Symphony/Avanto/Espree/
TRIO/7T/Aera/Skyra/Prisma/Sola/Vida

Installation and User's Guide
NUMARIS/4 VB15,VB17,VB19,VD11,VD13,VE11,VE12
NUMARIS/X VA11,VA20,VA40

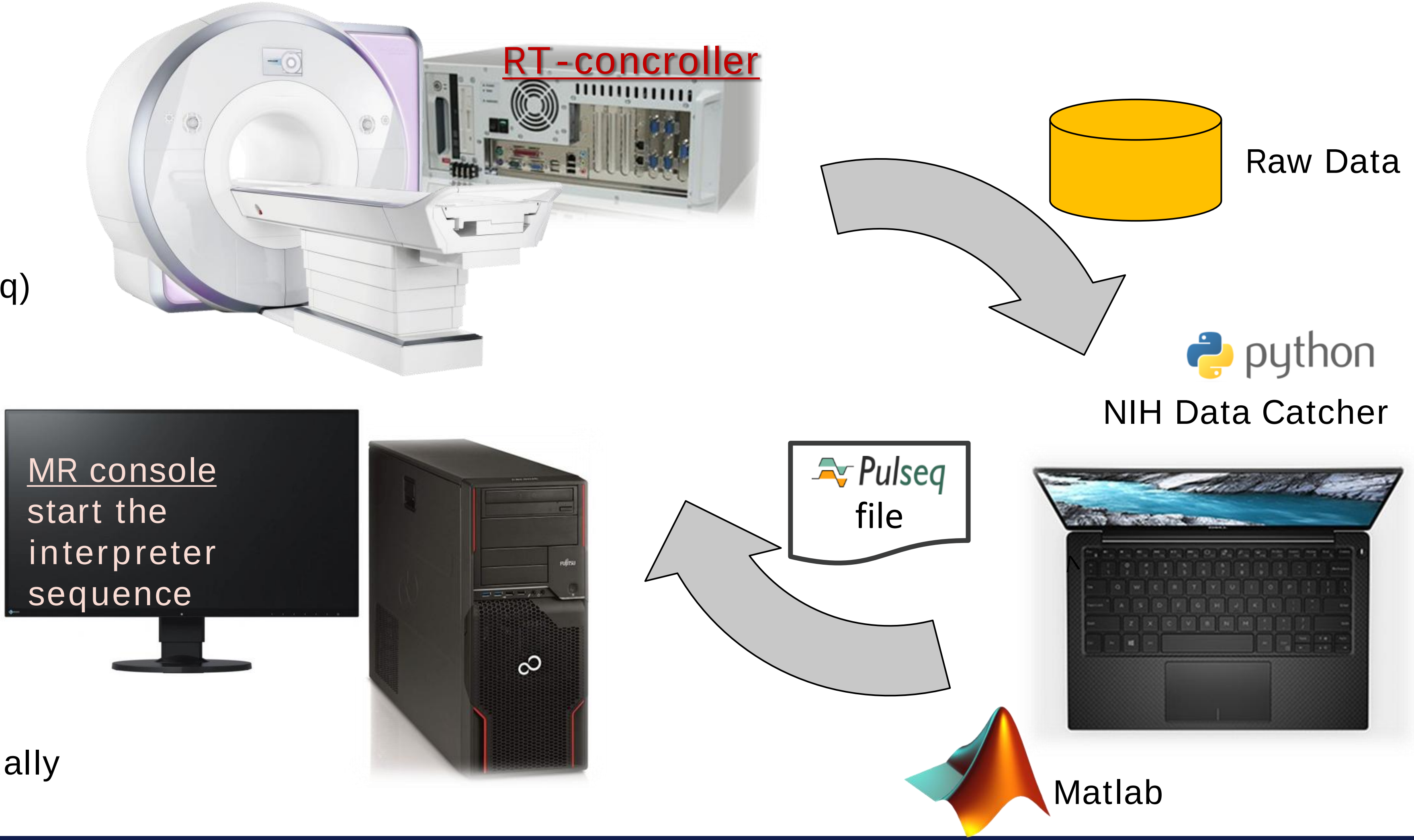
Maxim Zaitsev, Ph.D.

December 1, 2021

Magnetic Resonance Development and Application Center
MRDAC - Freiburg,
University Hospital Freiburg, Germany
Contact: Killianstr. 5a, D 79106 Freiburg,
email: info@mrdac.de, phone: +49 761 270-74120, fax -38310

Pulseq on Siemens scanners

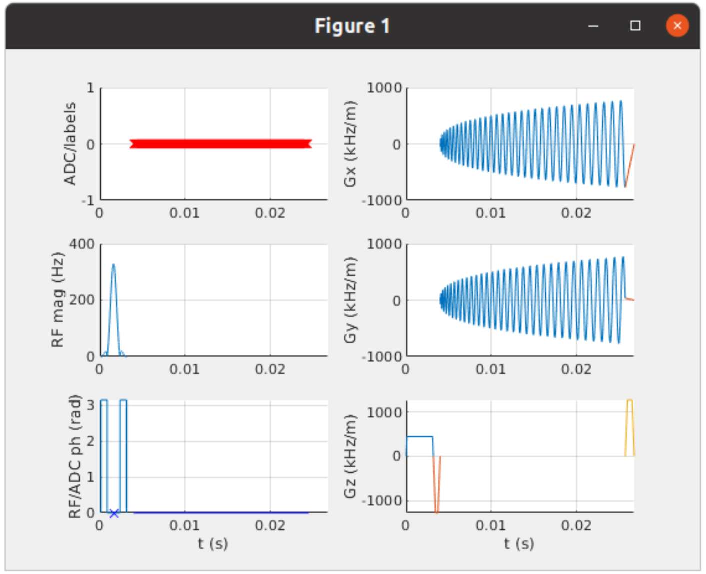
- *Optional initial step: connect your PC to the scanner*
- Save the .seq file on the scanner (e.g. external.seq)
- Run the interpreter_sequence on the scanner
- *Optional step: stream raw data to your with NIH_DataCatcher*
- or export raw data manually

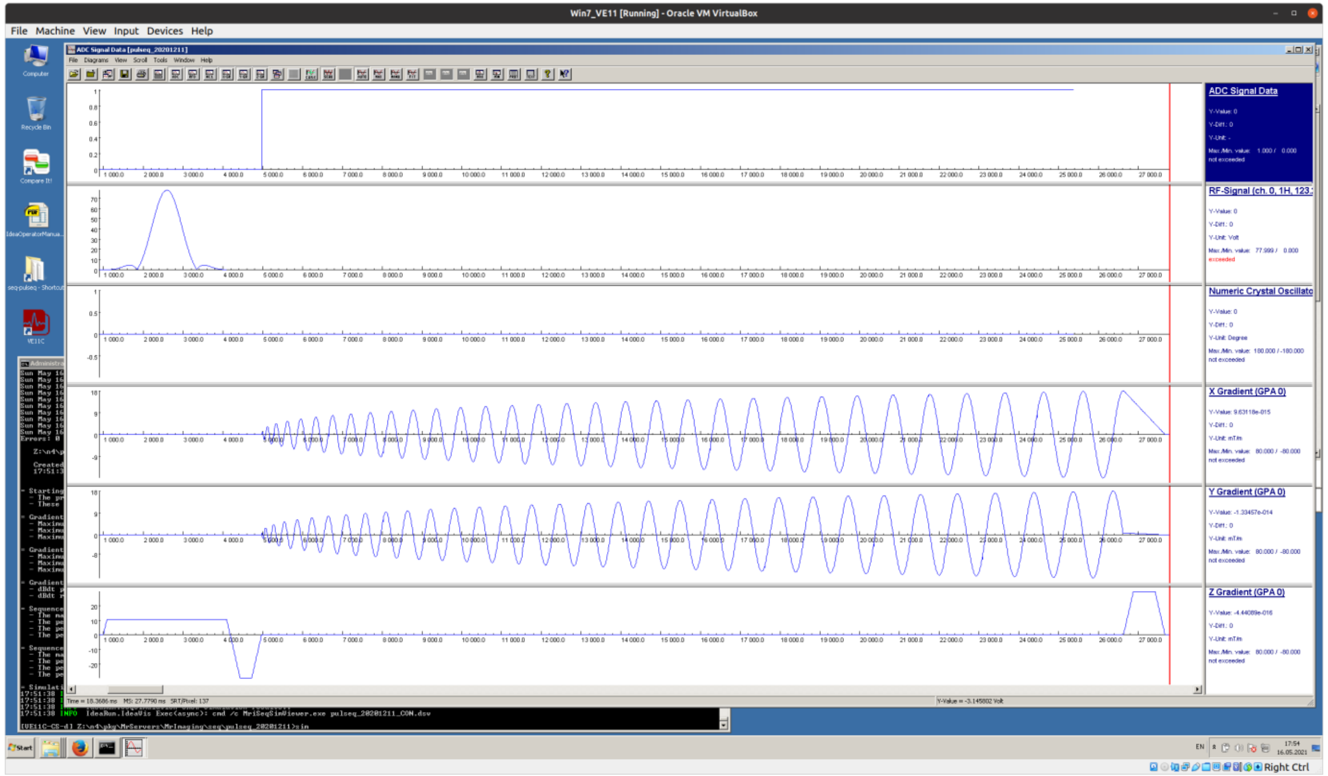


IDEA simulation with ***Pulseq***

Pulseq interpreter sequence can also be used with the Siemens' IDEA

1. Save your .seq file as %CustomerSeq%/Pulseq/external.seq
2. In the IDEA command run sim





Pulseq on Siemens platforms

- Over 40 C2P sites
- Works on all Numaris4 platforms (tested on vb15…ve12u) and numerous hardware platforms (Symphony, Trio, 7T, Connectom, Skyra, Prisma,…)
- Tested on selected NumarisX versions (xa11, xa20, xa30, xa40 …)
- Confirmed to work on Sola and Vida and 0.55T free.Max





Pulseq on the Internet

- Main Pulseq site & Matlab Toolbox:
  https://pulseq.github.io
- Python Pulseq Toolbox:
  https://github.com/imr-framework/pypulseq
- MRI Together live demo page on GitHub:
  https://github.com/pulseq/pulseqMRI_Together/
- Live raw data and results of the MRI Together demo on Dropbox:
  https://www.dropbox.com/sh/i7f1gpwyigdugps/AACd2jQJg_WjoTY2nqh7O8IHa?dl=0

Pulseq in the press

- De-facto standard for CEST collaboration
- Several MRM papers rely entirely on Pulseq
- Numerous ISMRM abstracts


1 of 14 Pulseq-CEST: Towards multi-site multi-vend... mrm.28825.pdf 61,4%

Received: 11 January 2021 | Revised: 9 April 2021 | Accepted: 12 April 2021
DOI: 10.1002/mrm.28825

FULL PAPER

Magnetic Resonance in Medicine

Pulseq-CEST: Towards multi-site multi-vendor compatibility and reproducibility of CEST experiments using an open-source sequence standard

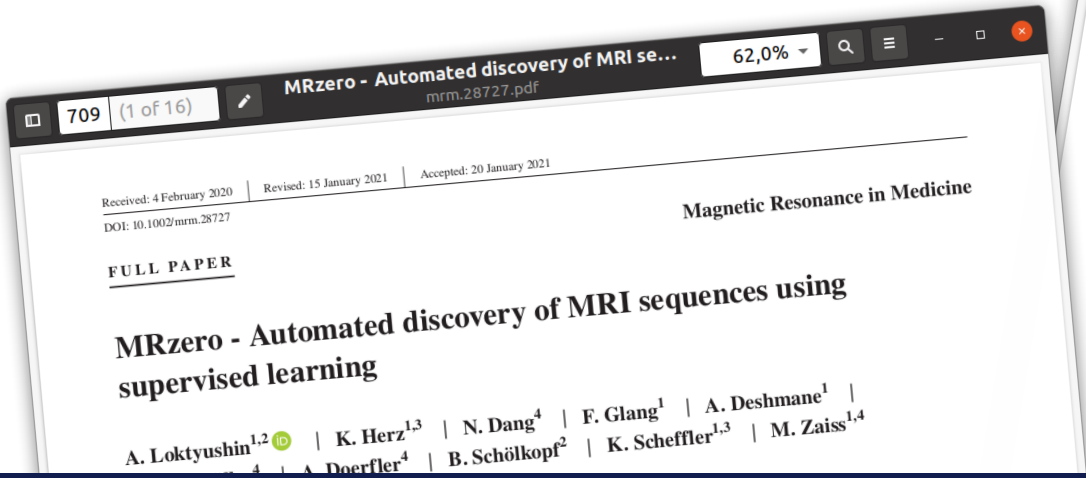
Kai Herz[1,2] | Sebastian Mueller[1,2] | Or Perlman[3] | Maxim Zaitsev[4] | Linda Knutsson[5,6] | Phillip Zhe Sun[7] | Jinyuan Zhou[6] | Peter van Zijl[6,8] | Kerstin Heinecke[9] | Patrick Schuenke[9] | Christian T. Farrar[3] | Manuel Schmidt[10] | Arnd Dörfler[10] | Klaus Scheffler[1,2] | Moritz Zaiss[1,10]

[1]Magnetic Resonance Center, Max Planck Institute for Biological Cybernetics, Tuebingen, Germany
[2]Department of Biomedical Magnetic Resonance, University of Tuebingen, Tuebingen, Germany
[3]Athinoula A. Martinos Center for Biomedical Imaging, Department of Radiology, Massachusetts General Hospital and Harvard Medical School, Charlestown, Massachusetts, USA
[4]High Field MR Center, Center for Medical Physics and Biomedical Engineering, Medical University of Vienna, Vienna, Austria
[5]Department of Medical Radiation Physics, Lund University, Lund, Sweden
[6]Russell H. Morgan Department of Radiology and Radiological Science, Johns Hopkins University School of Medicine, Baltimore, Maryland, USA
[7]Yerkes Imaging Center, Yerkes National Primate Research Center, Emory University, Atlanta, Georgia, USA
[8]F.M. Kirby Research Center for Functional Brain Imaging, Kennedy Krieger Institute, Baltimore, Maryland, USA
[9]Physikalisch-Technische Bundesanstalt, Braunschweig and Berlin, Germany
[10]Department of Neuroradiology, Friedrich-Alexander Universität Erlangen-Nürnberg, University Hospital Erlangen, Erlangen, Germany

Correspondence
Kai Herz, Magnetic Resonance Center, Max Planck Institute for Biological Cybernetics, Max-Planck-Ring 11, 72076 Tuebingen, Germany.
Email: kai.herz@tuebingen.mpg.de

Funding information
Max Planck Society, German Research Foundation (ZA 814/5-1, SCHU 3468/1-1, SFB 1340 project C03, and Reinhart Koselleck project DFG SCHE 658/12); National Institutes of Health (P41 EB015909, RO1EB019934, RO1 EB015032, R01 CA203873, and P41 EB024495); Swedish Research Council (2015-04170 and 2019-03637); Swedish Cancer Society (CAN 2015/251 and 2018/550); and Europ...

Purpose: As the field of CEST grows, various novel preparation periods using different parameters are being introduced. At the same time, large, multisite clinical studies require clearly defined protocols, especially across different vendors. Here, we propose a CEST definition standard using the open *Pulseq* format for a shareable, simple, and exact definition of CEST protocols.
Methods: We present the benefits of such a standard in three ways: (1) an open database on GitHub, where fully defined, human-readable CEST protocols can be shared; (2) an open-source Bloch-McConnell simulation to test and optimize CEST preparation periods in silico; and (3) a hybrid MR sequence that plays out the CEST preparation period and can be combined with any existing readout...
Results: The exact definition of the CEST...
the flexible simulati...


709 (1 of 16) MRzero - Automated discovery of MRI se... mrm.28727.pdf 62,0%

Received: 4 February 2020 | Revised: 15 January 2021 | Accepted: 20 January 2021
DOI: 10.1002/mrm.28727

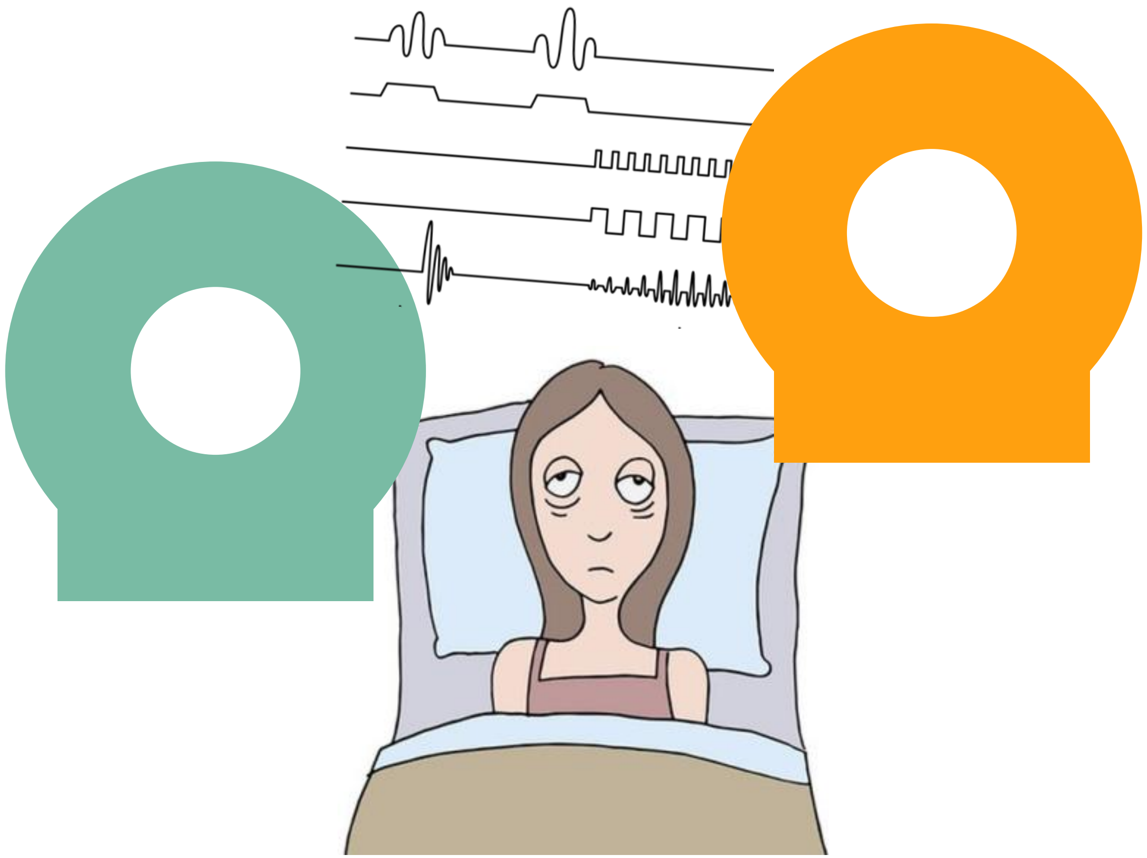
Magnetic Resonance in Medicine

FULL PAPER

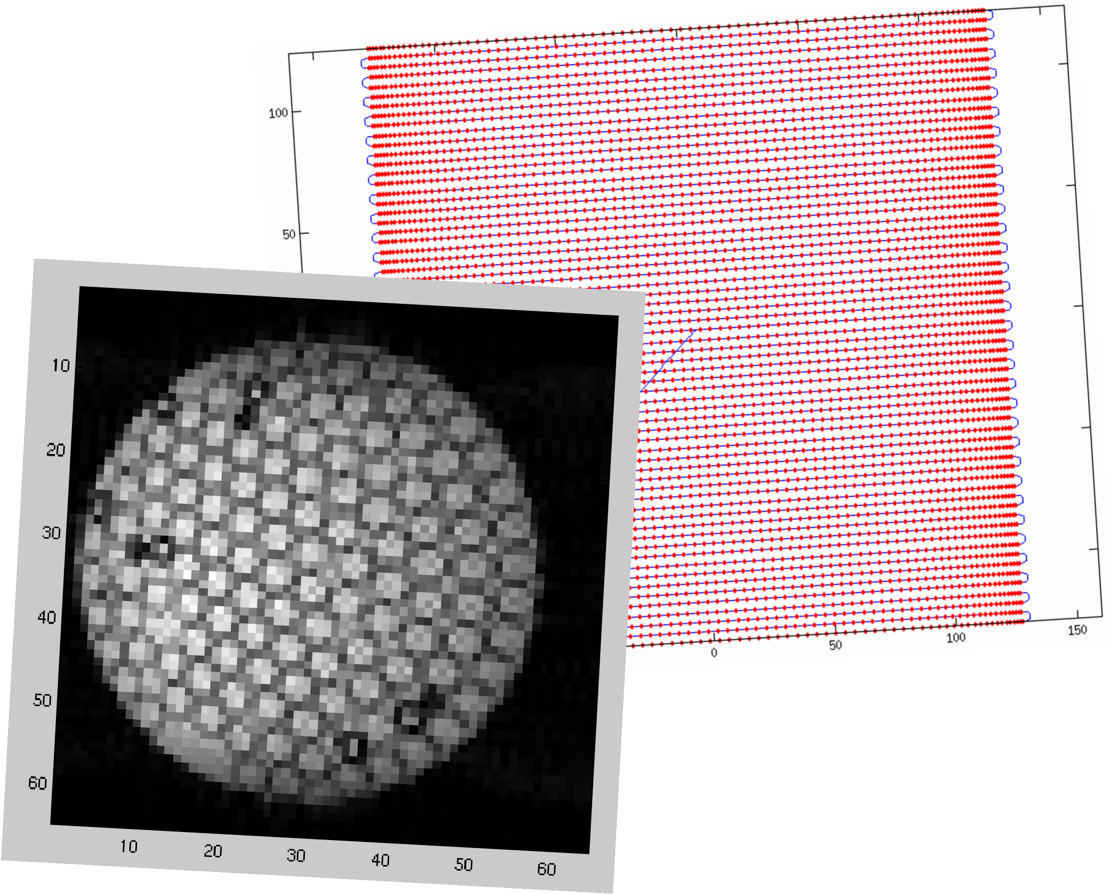
MRzero - Automated discovery of MRI sequences using supervised learning

A. Loktyushin[1,2] | K. Herz[1,3] | N. Dang[4] | F. Glang[1] | A. Deshmane[1] | B. Schölkopf[2] | K. Scheffler[1,3] | M. Zaiss[1,4] | ... A. Doerfler[4] | ...

Pulseq – that's the way you do it!



...dream of a sequence in the morning...



...check the images in the afternoon!

Gradient Echo from Scratch™ with *Pulseq*

- Fully-operational GRE sequence
  - Gradient spoiling
  - RF spoiling
  - PE refocusing
  - Multiple contrasts
  - …
- 64 lines of code including comments
- “zoom into” respective code sections in the following slides

/home/zaitsev/range_software/pulseq/matlab/tutorial_MRI_Together/seq/seq01_GradientEcho.m*

```matlab
% set system limits
sys = mr.opts('MaxGrad', 28, 'GradUnit', 'mT/m', ...
    'MaxSlew', 150, 'SlewUnit', 'T/m/s', ...
    'rfRingdownTime', 20e-6, 'rfDeadTime', 100e-6, 'adcDeadTime', 10e-6);

% basic parameters
seq=mr.Sequence(sys);           % Create a new sequence object
fov=256e-3; Nx=256; Ny=Nx;      % Define FOV and resolution
alpha=10;                       % flip angle
sliceThickness=3e-3;            % slice
TR=21e-3;                       % TR, a single value
TE=4e-3;                        % TE
%TE=[4 9 15]*1e-3;               % optionally give a vector here to have multiple TEs (e.g. for field mapping)
% TODO: change to multiple contrasts, change order of loops...

% more in-depth parameters
rfSpoilingInc=117;              % RF spoiling increment
rfDuration=3e-3;
roDuration=3.2e-3;              % not all values are possible, watch out for the checkTiming output

% Create alpha-degree slice selection pulse and corresponding gradients
[rf, gz, gzReph] = mr.makeSincPulse(alpha*pi/180,'Duration',rfDuration,...
    'SliceThickness',sliceThickness,'apodization',0.42,'timeBwProduct',4,'system',sys);

% Define other gradients and ADC events
deltak=1/fov; % Pulseq default units for k-space are inverse meters
gx = mr.makeTrapezoid('x','FlatArea',Nx*deltak,'FlatTime',roDuration,'system',sys); % Pulseq default units for gradient amplitudes are 1/Hz
adc = mr.makeAdc(Nx,'Duration',gx.flatTime,'Delay',gx.riseTime,'system',sys);
gxPre = mr.makeTrapezoid('x','Area',-gx.area/2,'system',sys); % if no 'Duration' is provided shortest possible duration will be used
phaseAreas = ((0:Ny-1)-Ny/2)*deltak;

% gradient spoiling
gxSpoil=mr.makeTrapezoid('x','Area',2*Nx*deltak,'system',sys);      % 2 cycles over the voxel size in X
gzSpoil=mr.makeTrapezoid('z','Area',4/sliceThickness,'system',sys); % 4 cycles over the slice thickness

% Calculate timing (need to decide on the block structure already)
delayTE=ceil((TE - gz.fallTime - gz.flatTime/2 ...
    - mr.calcDuration(gx)/2)/seq.gradRasterTime)*seq.gradRasterTime;
delayTR=ceil((TR - mr.calcDuration(gz) - max(mr.calcDuration(gxPre,gzReph),delayTE) ...
    - mr.calcDuration(gx))/seq.gradRasterTime)*seq.gradRasterTime;
assert(all(delayTE>=mr.calcDuration(gxPre,gzReph)));
assert(all(delayTR>=mr.calcDuration(gxSpoil,gzSpoil)));

% initialize the RF spoling counters
rf_phase=0;
rf_inc=0;

% define sequence blocks
for i=1:Ny % loop over phase encodes
    for c=1:length(TE) % loop over TEs
        rf.phaseOffset=rf_phase/180*pi;
        adc.phaseOffset=rf_phase/180*pi;
        rf_inc=mod(rf_inc+rfSpoilingInc, 360.0);
        rf_phase=mod(rf_phase+rf_inc, 360.0);
        %
        seq.addBlock(rf,gz);
        gyPre = mr.makeTrapezoid('y','Area',phaseAreas(i),'Duration',mr.calcDuration(gxPre),'system',sys);
        %seq.addBlock(mr.makeDelay(delayTE(c)),gxPre,gyPre,gzReph);
        seq.addBlock(mr.align('left', mr.makeDelay(delayTE(c)),gyPre,gzReph,'right',gxPre)); % stitch the read prephase to the readout gradien
        seq.addBlock(gx,adc);
        gyPre.amplitude=-gyPre.amplitude; % better to use mr.scaleGrad(gyPre,-1);
        seq.addBlock(mr.makeDelay(delayTR(c)),gxSpoil,gyPre,gzSpoil)
    end
end
```

seq01_GradientEcho.m : part 1

1. Start by defining system limits

- Do not be too greedy
  - Reduce maximum amplitude if you plan to apply FOV rotations
  - Reduce slew rate to avoid peripheral nerve stimulation (PNS) or reduce sound pressure

2. Define the Sequence object

3. Define convenience high-level parameters

  - Remember: there will be no UI to define or modify them later

```
% set system limits
sys = mr.opts('MaxGrad', 28, 'GradUnit', 'mT/m', ...
    'MaxSlew', 150, 'SlewUnit', 'T/m/s', ...
    'rfRingdownTime', 20e-6, 'rfDeadTime', 100e-6, 'adcDeadTime', 10e-6);

% basic parameters
seq=mr.Sequence(sys);           % Create a new sequence object
fov=256e-3; Nx=256; Ny=Nx;      % Define FOV and resolution
alpha=10;                       % flip angle
sliceThickness=3e-3;            % slice
TR=21e-3;                       % TR, a single value
TE=4e-3;                        % TE
%TE=[4 9 15]*1e-3;               % optionally give a vector here to have multiple TEs (e.g. for field mapping)
% TODO: change to multiple contrasts, change order of loops...

% more in-depth parameters
rfSpoilingInc=117;              % RF spoiling increment
rfDuration=3e-3;
roDuration=3.2e-3;              % not all values are possible, watch out for the checkTiming output
```

seq01_GradientEcho.m : part 2

4. Define RF and gradient pulses

- Define objects on-by-one
- Some functions return sets of objects that work together well
- *Remember to provide 'system'*
- Pulseq units make life simple
  - Gradients are defined in Hz/m
  - K-space unit is 1/m

```
% Create alpha-degree slice selection pulse and corresponding gradients
[rf, gz, gzReph] = mr.makeSincPulse(alpha*pi/180,'Duration',rfDuration,...
    'SliceThickness',sliceThickness,'apodization',0.42,'timeBwProduct',4,'system',sys);

% Define other gradients and ADC events
deltak=1/fov; % Pulseq default units for k-space are inverse meters
gx = mr.makeTrapezoid('x','FlatArea',Nx*deltak,'FlatTime',roDuration,'system',sys); % Pulseq default units for
adc = mr.makeAdc(Nx,'Duration',gx.flatTime,'Delay',gx.riseTime,'system',sys);
gxPre = mr.makeTrapezoid('x','Area',-gx.area/2,'system',sys); % if no 'Duration' is provided shortest possible
phaseAreas = ((0:Ny-1)-Ny/2)*deltak;

% gradient spoiling
gxSpoil=mr.makeTrapezoid('x','Area',2*Nx*deltak,'system',sys);      % 2 cycles over the voxel size in X
gzSpoil=mr.makeTrapezoid('z','Area',4/sliceThickness,'system',sys); % 4 cycles over the slice thickness

% Calculate timing (need to decide on the block structure already)
delayTE=ceil((TE - gz.fallTime - gz.flatTime/2 ...
    - mr.calcDuration(gx)/2)/seq.gradRasterTime)*seq.gradRasterTime;
delayTR=ceil((TR - mr.calcDuration(gz) - max(mr.calcDuration(gxPre,gzReph),delayTE) ...
    - mr.calcDuration(gx))/seq.gradRasterTime)*seq.gradRasterTime;
assert(all(delayTE>=mr.calcDuration(gxPre,gzReph)));
assert(all(delayTR>=mr.calcDuration(gxSpoil,gzSpoil)));
```

5. Plan your block structure

6. Calculate timing

- Some consistency/validity checks do not hurt

seq01_GradientEcho.m : part 3

<u>7. Loop to populate the sequence</u>

- Use seq.addBlock()
- Update or create new objects as needed
  - For 3D sequences “constant” objects with fixed IDs improve calculation speed
- No limitations
  - Mind calculation time in Matlab
  - Size of the .seq file may become a problem (after 60MB scanner may become unstable)

```
% define sequence blocks
for i=1:Ny % loop over phase encodes
    for c=1:length(TE) % loop over TEs
        rf.phaseOffset=rf_phase/180*pi;
        adc.phaseOffset=rf_phase/180*pi;
        rf_inc=mod(rf_inc+rfSpoilingInc, 360.0);
        rf_phase=mod(rf_phase+rf_inc, 360.0);
        %
        seq.addBlock(rf,gz);
        gyPre = mr.makeTrapezoid('y','Area',phaseAreas(i),'Duration',mr.calcDuration(gxPre),'system',sys);
        %seq.addBlock(mr.makeDelay(delayTE(c)),gxPre,gyPre,gzReph);
        seq.addBlock(mr.align('left', mr.makeDelay(delayTE(c)),gyPre,gzReph,'right',gxPre)); % stitch the read
        seq.addBlock(gx,adc);
        gyPre.amplitude=-gyPre.amplitude; % better to use mr.scaleGrad(gyPre,-1);
        seq.addBlock(mr.makeDelay(delayTR(c)),gxSpoil,gyPre,gzSpoil)
    end
end
```

<u>8. Recommended: seq.checkTiming()</u>

<u>9. Optional: visualization and further checks</u>

Structure of the tutorial

Seq01: basic Gradient Echo

Seq02: multi-echo gradient echo (monopolar readout with a fly-back rewinder)

Seq03: bipolar multi-echo gradient echo

Seq04: segmented gradient echo (variants a,b,c gradually increase in complexity; for Seq04c nSeg=nPE is possible – initial echo-planar (EPI) implementation)

Seq05: fairly time-optimal EPI sequence

Seq06: a step from EPI towards arbitrary trajectories, such as spirals, etc…

<u>All sequences are derived from each other</u>

use text compare tool (e.g. MELD) to see point-wise changes e.g. seq01 to seq02…

Recon scripts

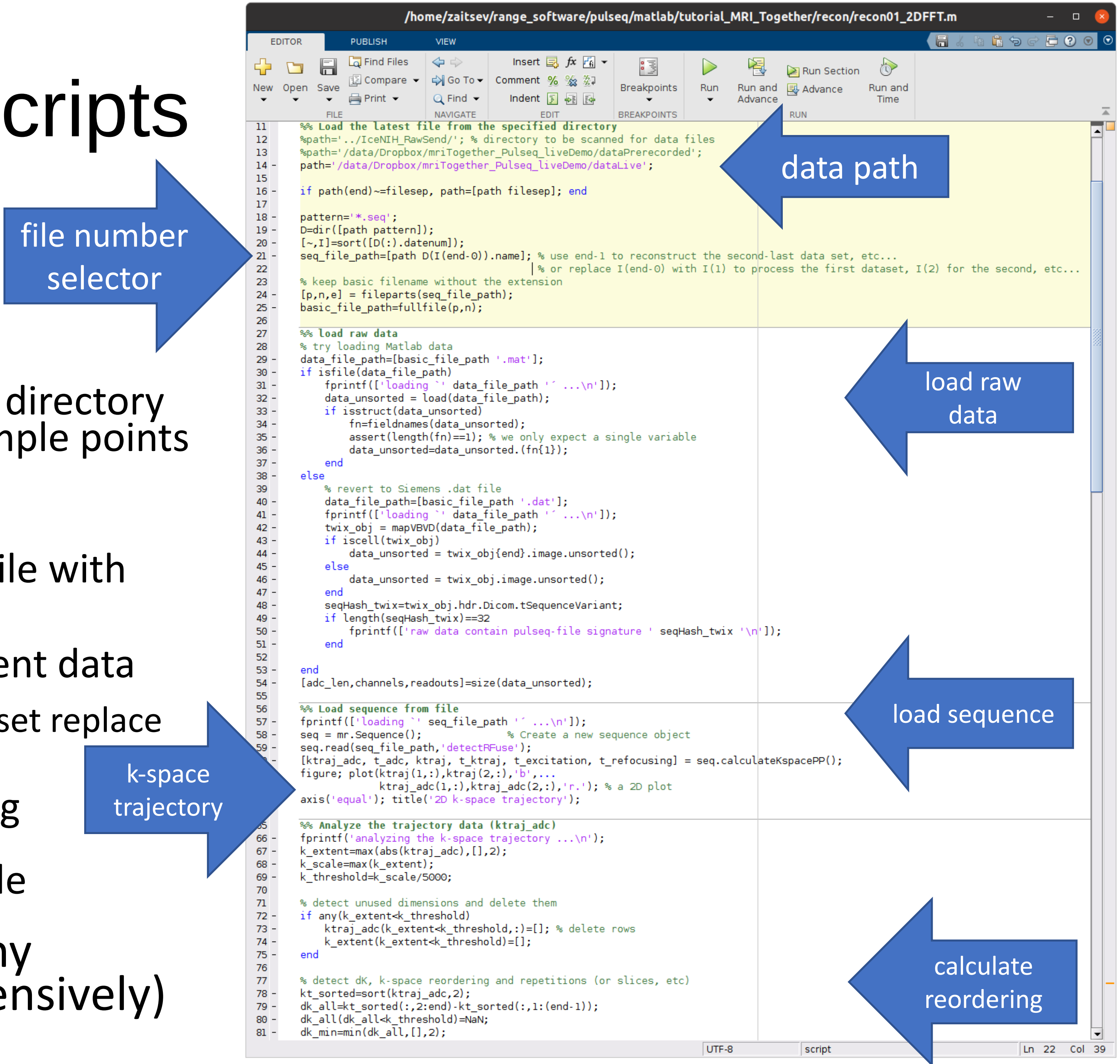
Recon01: fairly universal 2D FFT with automatic reordering detection
(see next slide)

Recon02: basic EPI reconstruction for ramp-sampling
contains some correction and compensation approaches

Recon03: basic 2D gridding followed by FFT

Pulseq – recon scripts

- All recon scripts are similar
- General logic
  - Expect data to be stored in a directory of choice (line 14 in this example points to data)
  - Raw data MUST always be accompanied with a Pulseq file with the identical name
  - By default load the most recent data
    - To load the second last data set replace ‘end-0’ with ‘end-1’ (line 21)
  - All data counters, data sorting and similar parameters are calculated from the Pulseq file
- 2D FFT can detect *almost* any reordering (will be used intensively)



Speaker name:
Prof. Maxim Zaitsev, Medical University of Vienna, Vienna, Austria, Earth

Conflicts of interest regarding this presentation:
Nothing to disclose

ESMRMB
European Society for Magnetic Resonance in Medicine and Biology