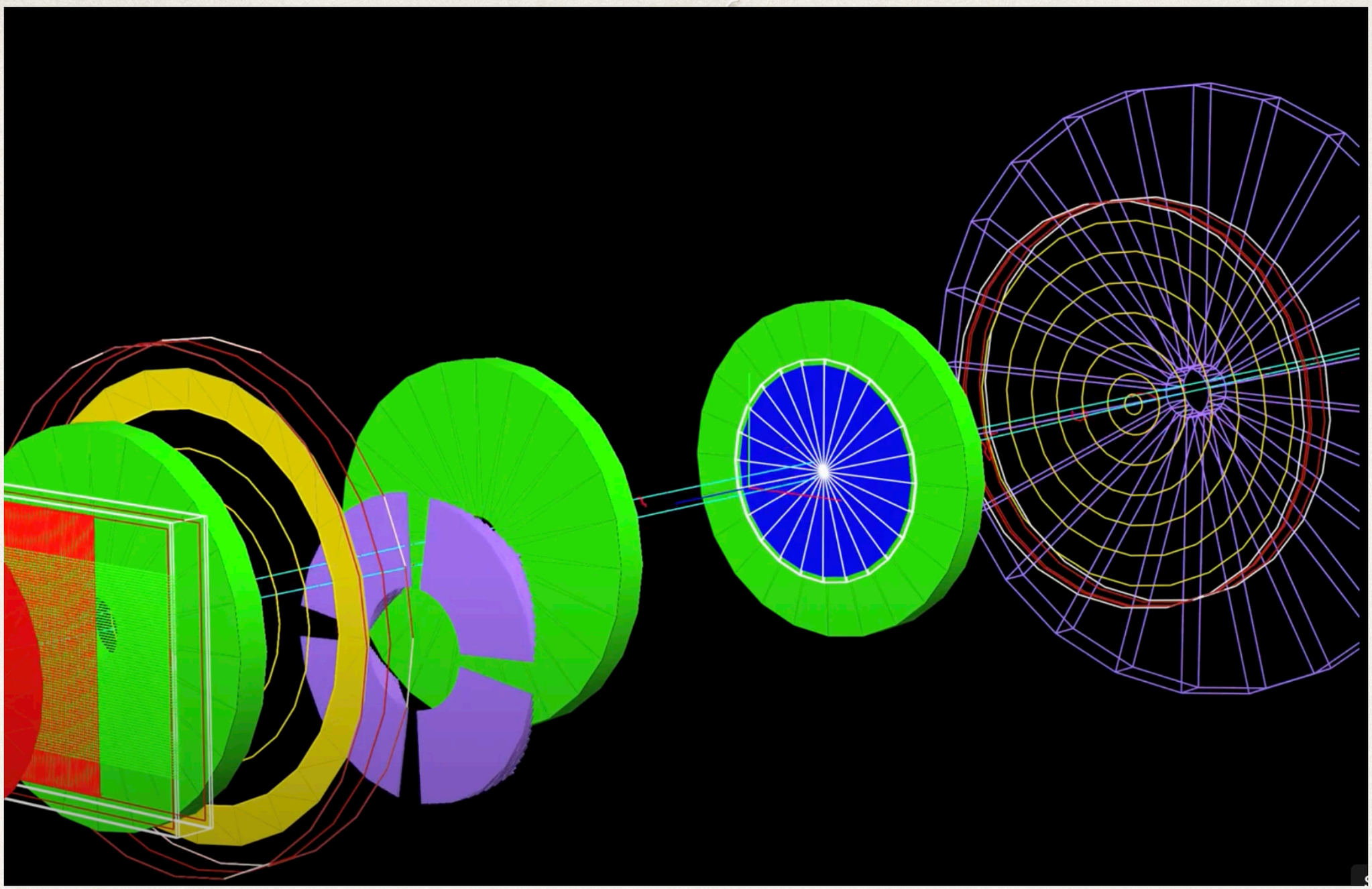
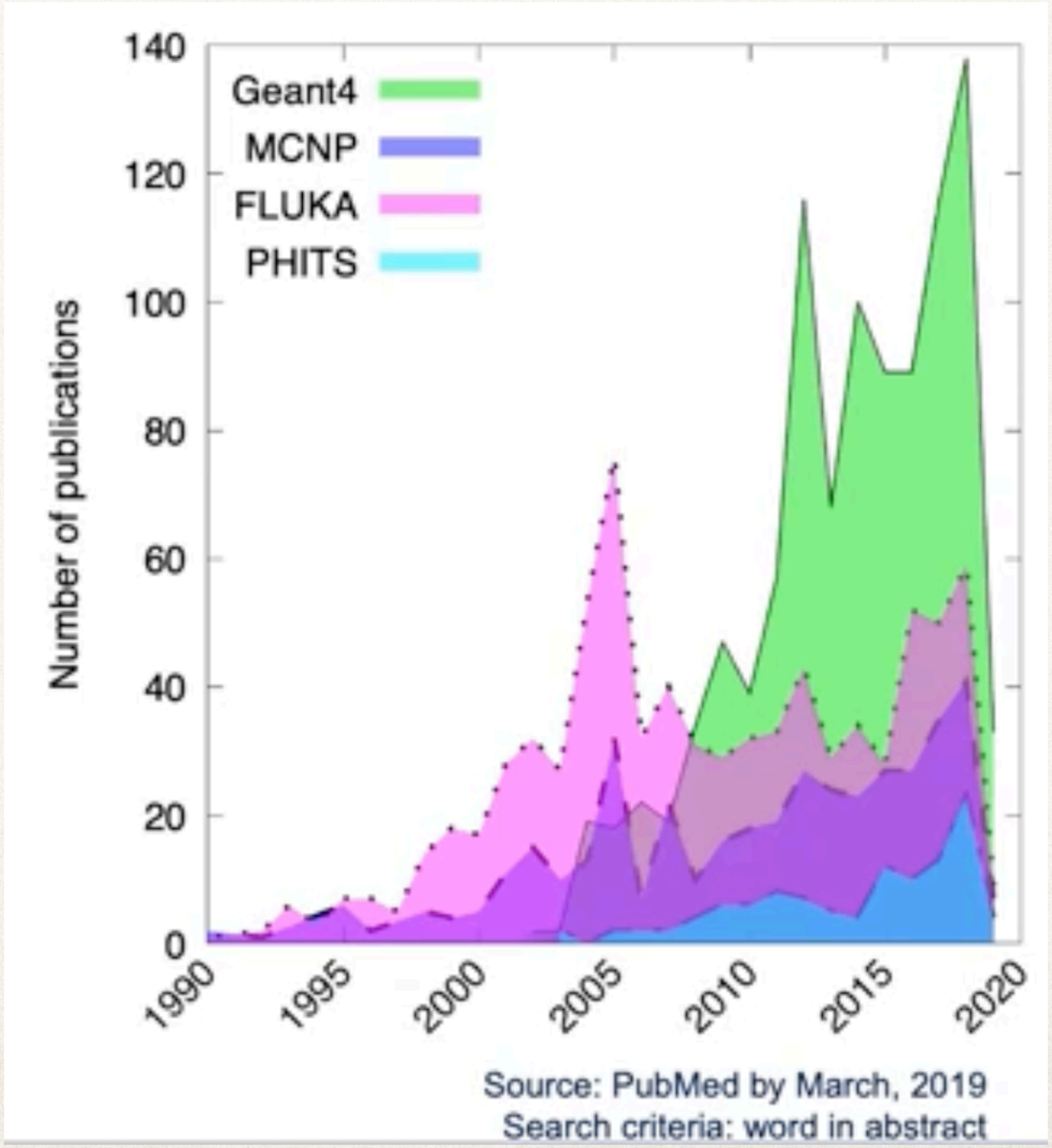


TOPAS

Alexandre Lindote

23 October 2023

MC toolkits used in medical physics



Philosophy

- No required knowledge of the underlying Geant4 Simulation Toolkit or any programming language: use "parameter control files"
- Users configure pre-built components (such as nozzles, patient geometry, dosimetry and imaging components)
- All aspects of the simulation, including all 4D behaviours, are controlled from a unique, TOPAS Parameter Control System (optional graphical interface)
- Includes a strong focus on "engineering-in" safety, employing a variety of techniques to make it harder for users to make mistakes
- Cannot be legally used as primary dose estimator for treatments, but can used as a secondary estimator
- Introductory videos and documentation available on their website

Capabilities

- Model X-ray and particle therapy treatment heads using simple (and more complex) geometrical forms

- Possible to model a patient geometry based on CT images (DICOM and other formats)

- Score energy, dose, fluence, etc

- Provides advanced graphics

- Fully four-dimensional (4D) to handle variations in beam delivery and patient geometry during treatment

Usage

- 2564 registered users at 638 institutions in 68 countries
- Early applications of TOPAS were in proton therapy
- Now applied in all areas of radiation therapy research
- Also suitable for some medical imaging applications
- Work ongoing to extend TOPAS to radiation biology (TOPAS-nBio)

Parameter control files

- Simple text files made up of lines of key/value pairs:
  Parameter_Type : Parameter_Name = Parameter_Value # Optional comment
- The order of the lines in the file does not matter

Parameter_Type tells TOPAS the type of data:

- `d` for Dimensioned Double (inc. units)
- `u` for Unitless Double
- `i` for Integer
- `b` for Boolean (must be in quotes)
- `s` for String (must be in quotes)
- `dv` for Dimensioned Double Vector
- similarly for `uv`, `iv`, `bv` and `sv`
  - Need to start with the number of elements
- Note: add ‘c’ to have the parameter in the UI

Complete info here

Parameter names have prefix conventions:

- `Ma/` for Materials
- `El/` for Elements
- `Is/` for Isotopes
- `Ge/` for Geometry Components
- `So/` for Particle Sources
- `Ph/` for Physics
- `Vr/` for Variance Reduction
- `Sc/` for Scoring
- `Gr/` for Graphics
- `Tf/` for Time Features
- `Ts/` for TOPAS overall control

More on parameter files

- One file may include separate files using
  includeFile = someOtherParameterFile
  - Use full path if file is in a different directory
  - You can include multiple files, or have a daisy-chain

- Parameters can be overridden or redefined
  Ge/IonChamber/Layer2/Foil/HLZ = Ge/IonChamber/Layer2/Foil/HLZ * 1.2
  Ge/IonChamber/Layer2/Foil/HLZ = inheritedValue * 1.2

- These features make it easy to start multiple jobs
  includeFile = MostOfMySettings.txt
  Ts/Seed = 1 # Set this differently for each of Job1, Job2, Job3, ...
  Sc/MyScorer/OutputFile = "Job1Output" # Set this differently for each of Job1, Job2, Job3,…

- There are default parameters, which you can modify in your files
  - See full list with description here

Run modes

- Fixed Time Mode
  - Single run, no time variation
    i:So/MySource/NumberOfHistoriesInRun = 100
- Sequential Time Mode
  - Several runs at fixed time intervals
    d:Tf/TimelineStart = 0. s # defaults to zero
    d:Tf/TimelineEnd = 10. s # must be larger than TimelineStart
    i:Tf/NumberOfSequentialTimes = 100 # defaults to 1
  - At each interval, the source will generate the set number of histories
  - You can have this number linked to the time interval
- Random Time Mode
  - If you want to have an idea of the time behaviour but the full sequence takes too long
  - Randomly sampled runs in a given time interval
    b:Tf/RandomizeTimeDistribution = "True" # defaults to "False"
    d:Tf/TimelineStart = 0. # defaults to zero
    d:Tf/TimelineEnd = 10. # must be larger than TimelineStart
    i:So/MySource/NumberOfHistoriesInRandomJob = 1000 # defaults to 100

More run control options

- Multithreading (default is only 1 core)
  i:Ts/NumberOfThreads = 4 # defaults to 1
  - Set to 0 to use all available cores
- Random number seed
  i:Ts/Seed = 12345 # default is 1
  - It is possible to save and reuse a seed from a problematic event
    i:Ts/FindSeedForHistory = 9998
    # defaults to -1, meaning do not activate this feature
  - This will output the seed to the screen and create a file
    TopasSeedForRun_0_History_9998.txt
  - You can then use this to start a run from this event
    s:Ts/SeedFile = "TopasSeedForRun_0_History_9998.txt"

Materials

- Topas has pre-defined materials (see here)
- Define a new material from elements

sv:Ma/Air/Components = 4 "Carbon" "Nitrogen" "Oxygen" "Argon" # names of elements
uv:Ma/Air/Fractions = 4 0.000124 0.755268 0.231781 0.012827 # fractions of elements
d:Ma/Air/Density = 1.2048 mg/cm3
d:Ma/Air/MeanExcitationEnergy = 85.7
s:Ma/Air/DefaultColor = "lightblue"

- Define a new material from existing materials

b:Ma/MyMixture/BuildFromMaterials = "True"
sv:Ma/MyMixture/Components = 2 "G4_WATER" "Air"
uv:Ma/MyMixture/Fractions = 2 0.5 0.5
d:Ma/MyMixture/Density = 0.5 g/cm3

  - Note that here we used one of the GEANT4 materials

- Make sure to give a unique name to your material

Geometry components

- Topas follows the GEANT4 paradigms
  - There is a special volume: “World”
  - New volumes are placed inside a parent volume
  - You must provide a translation and a rotation
  - You can also provide a colour (default is the material colour)
  - Use TsBox, TsCylinder and TsSphere instead of G4Box, G4Tubs and G4Sphere (for voxelisation)
    - These can be easily divided in bins

```
s:Ge/MyComponent/Parent = "World"
s:Ge/MyComponent/Type = "TsBox"
d:Ge/MyComponent/HLX=5. m #  Half Length
d:Ge/MyComponent/HLY=5. m
d:Ge/MyComponent/HLZ=5. m
s:Ge/MyComponent/Material = "Air"
d:Ge/MyComponent/TransX=0.0 cm # defaults to 0
d:Ge/MyComponent/TransY=0.0 cm # defaults to 0
d:Ge/MyComponent/TransZ=0.0 cm # defaults to 0
d:Ge/MyComponent/RotY=0.0 deg # defaults to 0
d:Ge/MyComponent/RotZ=0.0 deg # defaults to 0
:Ge/TestBox/XBins    = 3
i:Ge/TestBox/YBins    = 4
i:Ge/TestBox/ZBins    = 5
s:Ge/MyComponent/Color = "red"
# "Solid", "Wireframe" or "FullWireFrame"
s:Ge/MyComponent/DrawingStyle = "Solid"
i:Ge/MyComponent/VisSegsPerCircle = 100
b:Ge/MyComponent/Invisible = "True"
```

Specialised components

See the SpecialComponents folder in examples (more details here)

- Range Modulator Wheel → TsRangeModulator
- Propeller → TsPropeller
- Ridge Filter → TsRidgeFilter
- Multi Wire Chamber → TsMultiWireChamber
- Jaws → TsJaws
- Multi Leaf Collimator → TsMultiLeafCollimator
- Doubly Diverging Multi Leaf Collimator → TsDivergingMLC
- CAD (Computer Aided Design) → TsCAD
- Aperture → TsAperture
- Compensator → TsCompensator
- BrachyApplicator → TsBrachyApplicator
- Pixelated box → TsPixelatedBox
- geometry_eyemodel → TsEye
- geometry_eyeplaque → TsEyePlaque

Patient components

More details here

- Patient in DICOM Format → TsDicomPatient
  - You can even specify a 4DCT, by having *DicomDirectory* change under control of a Time Feature
  - Not only CT files, but also MRI and Ultrasound
  - Supports RT Structure Sets (information about identified organs, tumours, etc.)
  - Supports RTDOSE files (creates a scoring grid following this information)
- Patient in ImageCube Format (handles XCAT, XiO, MaterialTagNumber and more) → TsImageCube (handles XCAT, XiO and more)
- In both cases it is necessary to define an image to material conversion scheme

Scorers

- Volume Scorers or Surface Scorers
  - Generally scorers output overall quantities over many particles/events, but you can set per-particle information too
- There are several pre-programmed quantities you can use
  - s:Sc/MyScorer/Quantity = “DoseToMedium"
  - Other quantities available here
- For Volume Scorers you must indicate the relevant component
  - s:Sc/MyScorer/Component = "Phantom"
- For Surface Scorers the available quantities are SurfaceCurrent, SurfaceTrackCount and PhaseSpace
  - You must indicate the component and the relevant Surface name
    s:Sc/MyScorer/Surface = “Phantom/ZMinusSurface"

Scorers continued

- You can easily add filters to score only particles with certain properties (origin, charge, energy, etc.)
- Output file and type (more info here):
  - *s:Sc/MyScorer/OutputFile = "~/SomeSubdirectory/myOutputFileName"*
  - *s:Sc/MyScorer/OutputType = "csv"*
    "csv", "binary", "Root", "Xml" or "DICOM"
  - The DICOM output uses RTDOSE format
  - If using Root or Xml you must define the histogram to use
    i:Sc/MyScorer/HistogramBins = 100 # number of bins
    d:Sc/MyScorer/HistogramMin = 0. MeV # with unit appropriate to scored quantity
    d:Sc/MyScorer/HistogramMax = 100. MeV # with unit appropriate to scored quantity
  - Additional output control options:
    b:Sc/MyScorer/OutputToConsole = "True" # control whether output is also dumped to console
    s:Sc/MyScorer/IfOutputFileAlreadyExists = "Increment" # "Exit", "Overwrite" or "Increment"
- When scoring in Dividable Components (TsBox, TsCylinder or TsSphere), you have many binning options. By default, binning will match the divisions of the volume. So if you have divided the component, the score will be divided in the same manner. But you can also define a different binning just for the scorer.

Online resources

- Installation is trivial (I tried it on macOS and Linux)
  - You just download and unzip the TOPAS folder
- TOPAS has a very lively User Forum
  - You'll need to register to access it, and attend an introductory session (this is actually very useful)
- Step by step User Guide, with detailed explanation on all the simulation aspects (geometry, sources, scorers, control, etc.)
- Introductory videos from the TOPAS creators available here

Hands-on exercises

Exploring examples

- Open the *QtShapeTest.txt* (in *examples/Basic*) and let's look at all the basic available shapes, and the meaning of their various parameters
  - Feel free to change parameters on each one to see what happens
- Open the *QtTest* text file, run the simulation and let's play with the available parameters
  - Let's create some a new volume and a new scorer
  - Once we're happy we can save our new setup to a txt file and use it in the future!
  - We can even do this in multiple steps:
    change the name of the new file, open it and create a new source
- We're used to a standard (GEANT4) color scheme for particle trajectories, but TOPAS allows several different schemes. Let's try a couple from the *examples/Graphics* folder
  - This may be very helpful to test and diagnose problems you may have in your simulation

More examples

- Explore some of the components in the SpecialComponents folder and play with the available parameters
- Use divided volumes to create simple phantoms
  - By default a volume has only one material, but we can divide it and attribute different materials to each voxel
  - This is actually how TOPAS handles Dicom and other CAD-like geometries
  - Let's check the *VoxelMaterialsInDividedComponents.txt* (in the Basic examples folder)

Multiple jobs example

- TOPAS makes it easy for you to start multiple jobs with the same configuration but different seeds
- This is a simple use of the possibility to include files
  - One primary file with all our configurations
  - Then one very simple file per job
  - We can even set different output files for each job
- This is illustrated in the BatchJob#.txt (Basic examples)
  - Run job 1 with 10000 events and job 2 with 20000

Patient materials example

- Assign materials using an ImageToMaterial converter
  - E.g. Schneider for CT numbers (Houndsfield Units)
  - Let's look at a simple example: *Implant.txt* in the Patient examples folder
    - You'll need to unzip DICOM_Box.zip before running
    - The phantom is a simple water box with some bone voxels inside
    - It uses the special TsDicomPatient type of solid
  - This is also a simple example of using parallel worlds to add volumes inside a patient
    - Useful for simulating brachytherapy treatments

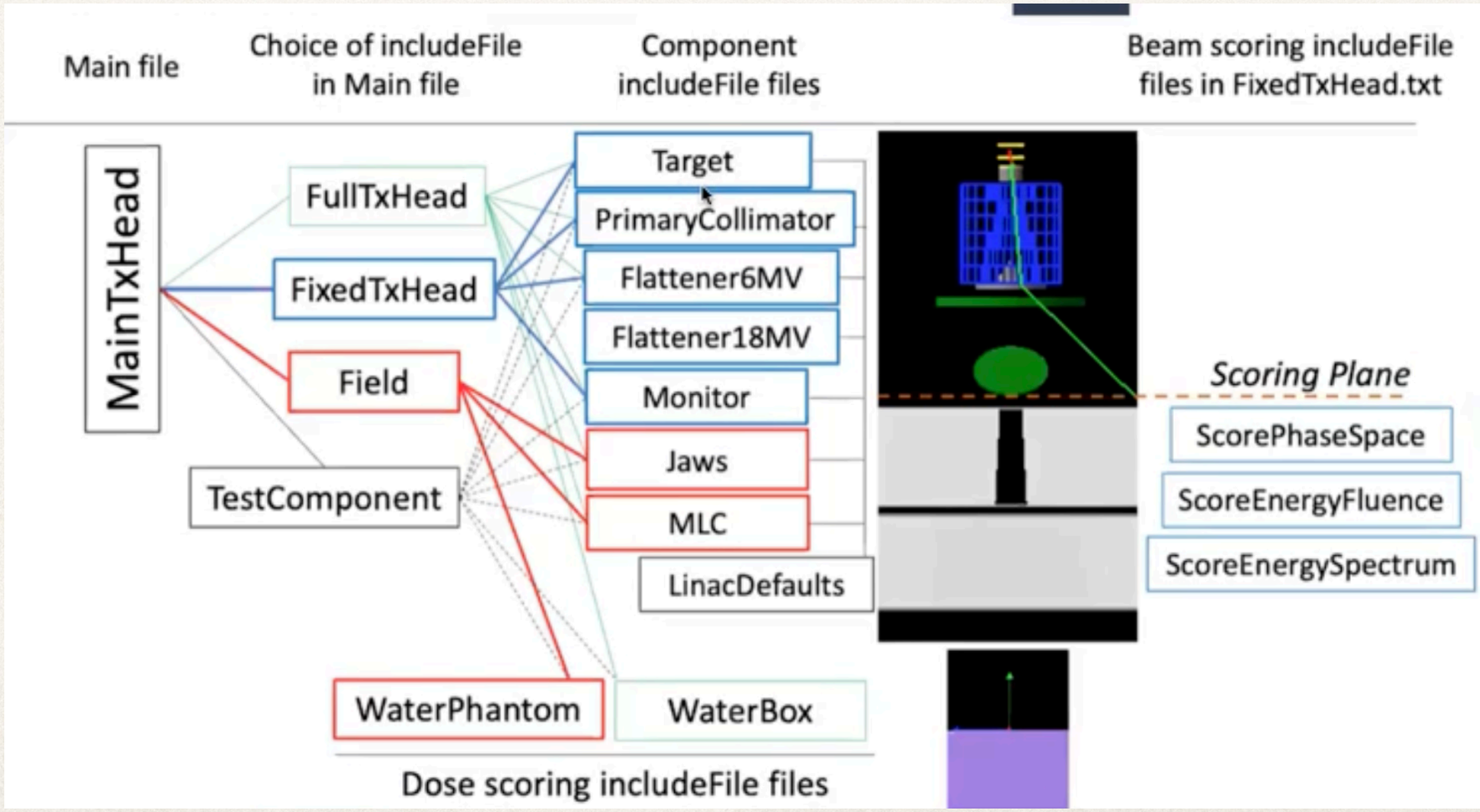
Simple exercise

- Let's create a very simple simulation
  - Create a water cylinder in your geometry, this will be our phantom (r=10 cm, h=50 cm)
  - Voxelize your phantom (8x8x8) and add a scorer to register the dose in each voxel
  - Create a small plastic sphere (r=1 cm) some 50 cm away from the side of your phantom
  - Create two isotropic gamma sources (1.17 and 1.33 MeV) in this sphere — you can start by placing them in the center, or homogeneously distributed
  - Run some histories and let's quickly analyse the output of the scorer

More complex examples

LINAC

- Simulates a linear accelerator treatment head (Siemens ONCOR machine)
- examples/MVLinac



Simulation from DICOM files

- TOPAS reads DICOM CT
  - Generates 3D voxelized structure filled with materials (HU conversion)
  - Places the CT center with respect to its mother volume
- TOPAS reads DICOM RTDOSE
  - To generate the scoring grid
- TOPAS reads DICOM RTSTRUCT
  - To filter the results of the simulation to the structures of interest
- TOPAS writes DICOM RTDOSE
  - Output of the simulation
- TOPAS does not read DICOM RTPLAN (yet)

Simulation from files

- *ViewAbdomen.txt* example
- Conversion from CT HU (attenuation coefficients) to materials required
- Additional file (*HUtoMaterialSchneider.txt*) including:
  - Relation HU - mass density (g/cm3) for tissues
  - Using method by Schneider et al. (2000), PMB 45 459
    - ρ = offset + factor * (factorOffset + HU)
    - Note that an additional density correction factor is required (between the TPS and GEANT4)
  - Materials used for HU 'built' from 13 elements:
    H, C, N, 0, Mn, P, S, CI, Ar, Ca, Na, K, Ti
    - List of 26 materials ("SchneiderHUToMaterialSections")
    - Proportions of elements for each one of the materials
- XCAT.txt for reading an alternative format

Time features

- Step-by-step exercise to create a simple rotation
- Patient with a rotating beam (*PatientInIEC_2_time_feature.txt*)
- Range modulator — rotating wheel and different beam intensity
- More advanced — Nozzle