Staying on Script:
Automating the Plan Check Process and Beyond

Grayden MacLennan, MBA, MSM, MS, CMD

Disclosures

• No personal disclosures

• This presentation mentions a few commercial software packages
About the Presenter

• Grayden MacLennan, MBA, MSM, MS, CMD

• Unusual career path
  – Pathology Lab Tech
  – IT Guy & Webmaster
  – Medical Imaging Software Trainer
  – Medical Dosimetrist
About the Presenter

- Computer enthusiast, but not a natural born programmer

- Career goal:
  - Learn enough coding to automate the tedious parts of medical dosimetry (and maybe physics)
  - Make more time for the cool stuff

Roadmap for Today

- Learning news skills
- Basic skills for scripting
- Understanding data
- Learning how to access data
- What do you DO with the data?
- Scripting examples from several centers
- Looking for plan quality too
Learning a New Skill

- Today we’ll aim a bit lower
Learning a New Skill

• Don’t get intimidated
  – You don’t need to learn it all

• Remember the 80/20 rule
  – 80% of the results come from 20% of the skills

• What are the bare minimum skills needed?

Operating a Car

• Bare minimum skills
  – Acceleration
  – Braking
  – Steering

• Advanced topics
  – Turn signals
  – Road markings
  – Traffic laws
Computer Automation

• Simple and repetitive tasks done on a computer can often be automated

Scripts for Automation

• What is a script?
  – Series of precise instructions
  – Written in a way a computer can understand

• Basic concept:
  – **Find** some data
  – **Do** something with it
Plan Check Tasks

- Most plan check tasks are simple
  - Look up an important piece of data
  - Compare its value to an expected value (optional)
  - Report what was found

- Most plan check scripts are just these couple actions repeated again and again

What Data?

- Patient info
- Technical plan parameters
- Plan completeness
- Plan quality
- Data transfer integrity
- Etc...
Why Do This?

- AAPM TG-100
  - Application of risk analysis methods to radiation therapy quality management

- AAPM TG-275
  - Strategies for Effective Physics Plan and Chart Review in Radiation Therapy

This Has Been Discussed


The 80/20 of Scripting

- Huge portion of plan checking can be done with just a few core concepts
  1. Relational operations
  2. Conditionals
  3. Loops
  4. Structured data

Core Concepts

1. Relational operations
2. Conditionals
3. Loops
4. Structured data
1. Relational Operations

- Compare two values to each other
  - < Less than
  - > Greater than
  - == Equal to
  - != Not equal to

- Two possible outcome of a relational test
  - True
  - False

1. Relational Examples

<table>
<thead>
<tr>
<th>Test</th>
<th>Result</th>
</tr>
</thead>
<tbody>
<tr>
<td>3 &lt; 10</td>
<td>True</td>
</tr>
<tr>
<td>Salary &gt; $100,000,000</td>
<td>False</td>
</tr>
<tr>
<td>Speaker == “Grayden”</td>
<td>True</td>
</tr>
<tr>
<td>Topic != “horticulture”</td>
<td>True</td>
</tr>
</tbody>
</table>
Core Concepts

1. Relational operations
2. Conditionals
3. Loops
4. Structured data

2. Conditionals

• True/False tests can guide behavior

if (hungry)
  eat
2. Conditionals

- Combine a conditional with a relational test...

```python
if (plan MU > 100000)
    print "hey now"
```

2. if/else Statements

- Alternative action can be specified with "else"

```python
if (feelinglucky)
    say "Go Sooners!"
else
    say "Go Longhorns!"
```
Core Concepts

1. Relational operations
2. Conditionals
3. Loops
4. Structured data

3. Loops

• If something needs to be done multiple times, a loop is what you need

• Two kinds of loops
  – “While” loops
  – “For” loops
3. “While” Loops

- Everything in the loop is done while the test condition is still true

    while (presentation not finished)  
    advance slide  
    talk about slide

3. “For” Loops

- To do something for each item in a set or list

    for each person in audience  
    check if conscious
Core Concepts

1. Relational operations
2. Conditionals
3. Loops
4. Structured data

4. Structured Data

• Data can be found in many forms
  – Basic types
  – Lists
  – Classes
4. Basic Data

- Basic types store one piece of information
  - **Number** (e.g. – Prescription dose)
  - **Text** (e.g. – Name)
  - **True/False** (e.g. – Plan approved)
    - Also called “Boolean”

4. Lists

- A way to reference **multiple** similar items
- Could potentially be a list of **other lists**
4. Classes

- A framework for wrapping together related pieces of information
- Creates **naming convention** for accessing data components
- Can create links to other classes
  - Allows construction of **data hierarchies**
- More of a road map than actual data itself

4. Hierarchy Example

- Folder structure is good example of hierarchy

  C:\Pictures\Memes\Cats\grumpy_cat.jpg
4. Hierarchy for Plans

- Plans have complex multilayered relationships

- "Drilling down"
  - A patient has treatment courses
    - A course has plans
      - A plan has beams
        - A beam has MLCs
          - An MLC has leaf positions
          - ...

But Why Scripting?
Scripting for Plan Checks

- Computers are good at well-defined repetitive tasks
- Classes provide a structure to clearly define the data that needs to be evaluated
  - Relationships between classes make it possible to walk through the data tree to select the right data
- Repetition is easy with loops and conditional statements

Where is the Data?

MacLennan 2018
Data Locations

- Data for a plan check is most often:
  - In the treatment planning system
  - In a database server
  - In a folder on a network drive

Data in a Plan Check

- Most common files examined in plan checks
  - DICOM and DICOM-RT
    - Image sets, most likely CT and DRR
    - RT Structure Sets (ROIs)
    - RT-Plan
    - RT-Dose
  - Documents
    - PDF
    - Text
Getting to the Data

• For whatever plan check task we want to perform, we need to figure out how to feed data into it

• Data may not follow a class structure while in its stored or retrieved state
  – It must be read from its source and loaded into the logical structure defined by classes

Reading Data Sources

• That wasn’t part of the deal!

  – We thought we only needed a handful of high level comparison and loop commands

  – Reading data sources uses lots of low level code and commands
High Level vs. Low Level

- A simple command can be broken successively into more and more detailed subcommands

**“High level” instructions**
- Open Door
- Grasp Doorknob
- Turn Doorknob

**“Low level” instructions**
- Excrete Ca$^{2+}$ from sarcoplasmic reticulum in Supinatoe Muscle

Using Middlemen

- High-level instructions are much easier for us
- We need a mechanism to **shield us** from having to know all the low level details

"Let’s just say I know a guy... who knows a guy... who knows another guy"
All About APIs

- API = Application Programming Interface
- Acts as a middleman to provide a more friendly way to interact with a system
- Defined by software makers to provide:
  - List of data **objects** that can be accessed
  - List of **actions** that can be performed

Data Access with APIs

- **Objects** are unique instances of a **class** that have already been loaded with data.
  - They follow the rules and structure of the class
- Dot notation is usually used for exploring the object’s data tree
  - Patient.Name.Last  →  Paulson
  - Patient.Name.First  →  Robert
Scripts in the Real World

Many Ways to Get Data
What Are You Using?

Samantha Hedrick, PhD, DABR
Emory Proton Therapy Center
Atlanta, GA

Emory’s Script

- Runs inside RayStation
  - Uses RayStation’s built-in scripting API
  - Written in Python

- Creates detailed plan summary
- Looks for common problems

- Output goes to a text file
Emory’s Approach

- Treatment Planning System
- Scripting API
- Scripting System

Emory: User Interface

- To use Emory’s script, open the Scripting tab in RayStation
  - Select the script and run it
- Final output will pop up in a Notepad window
Emory: Plan Info

- Printout includes basic information:
  - Patient name
  - Patient ID
  - Plan name
  - Treatment site
  - Treatment planning CT
  - CT to ED table name

Emory: Image check

- Compare CT’s acquisition energy to name of CT-ED table
  - Warn if mismatch

- Compare CT’s patient position to plan patient position
  - Warn if mismatch
Emory: Dose & Setup

- Checks that dose is calculated
- Check that dose grid size 3 mm or less
- Check that a CT marker POI exists
- Check that a couch ROI exists

Emory: Contour Check

- Loop through each ROI:
  - Check if ROI is empty
  - Check if ROI has density override
  - Check if ROI export enabled
- Print list of empty ROIs
- Print list of ROIs with overrides
- Print list of ROIs to be exported
Emory: Beamset Check

- Loop through each beamset in plan:
  - Identify if background dose or planned dose
  - Warn if no beams in beamset
  - Print:
    - Beamset name
    - Prescribed dose
    - Number of fractions
    - Dose grid size
    - Treatment room
    - Robustness parameters if used

Emory: Beam Info

- Loop through each beam in beamset:
  - Print info:
    - Beam number
    - Beam description
    - Snout name
    - Air gap
    - Beam dose
    - Avoidance ROI if applicable
    - Layer repainting setting
Emory: Beam Warnings

- Loop through each beam in beamset:
  - Report common problems as applicable:
    - Dose specification point missing
    - Aperture is misnamed
    - No POI matches beam isocenter location
    - No range shifter selected
    - Beam number does not match name
    - Air gap not appropriate for range shifter

Emory: Beam Warnings

- Loop through each beam in beamset:
  - Report common problems as applicable:
    - Gantry angle does not appear in beam name
    - Couch angle does not appear in beam name
    - Isocenter more than 5 cm from target ROI center
    - Beam or couch angle not achievable with selected treatment room
    - Repainting not selected if plan name includes “lung”
Emory: Proton Warnings

• Loop through each **PBS layer** in beam:
  – Report common problems as applicable:
    • Layer energy not deliverable
    • Layer contains only 1 PBS spot
    • Layer delivers less than minimum weight
    • Minimum spot weight not adjusted for repainting

What Are You Using?

Eric Ford, PhD, FAAPM
Patricia Sponseller, MS, CMD, RTT(R)(T)
University of Washington Medical Center
Seattle, WA
UW’s Script

- Runs inside Pinnacle
  - Uses Pinnacle’s built-in scripting API
  - Uses Pinnacle’s proprietary language
  - Currently being rewritten for RayStation

- Creates detailed plan summary
- Looks for common problems
- Prompts user to identify less common scenarios

- Output goes to a window in Pinnacle
UW: SBRT?

- Ask if the plan is SBRT

- If yes
  - Ensure selected vault is SBRT-capable
  - Dose grid must be smaller than usual

UW: Pacemaker?

- Ask if patient has pacemaker

- If yes
  - Verify that all beams are low energy
What Are You Using?

Hsiao-Ming Lu, PhD
Massachusetts General Hospital
Boston, MA
MGH’s Program

- Standalone program called planCheck
- Developed in-house
- Reads data from:
  - PDF plan printouts in R&V database
  - PDF MU 2\textsuperscript{nd} Check printout in R&V database
- Compares PDF data to:
  - Plan values entered into R&V database
  - Plan data held in “TPDB” ancillary database

MGH: Data Sources

- Treatment Planning System
- MU Second Check
- Record & Verify Database
- “TPDB”
- Scripting System
MGH: Standard Checks

- Basic checks applied to all plans
  - Prescription signed in R&V
  - Plan document signed in PDF
  - Any overdue treatments?
  - Dose coefficients calculated in Mosaiq?
  - Appropriate calculation algorithm selected?

MGH: Per-Type Checks

- planCheck defines several treatment classes based on treatment types and modalities
  - Each class has checks unique to it
  - Inheritance allows compiling of final set of checks based on class membership
MGH: Class Example

- Proton XiO patient class checks
  - CT calibrated?
  - Calibration appropriate?
  - Beams have same name and snout?
  - Isocenter matching between beams?
  - Each beam has a DRR?
  - Many many more...

planCheck Interface
What Are You Using?

Fazal Khan, CMD
Miami Cancer Institute
Baptist Health South Florida
Miami, FL

MacLennan 2018 75

MCI: Scripts

• Standalone scripts running in Python interpreter

• Reads plan data directly from DICOM files exported by TPS to a folder

• API
  – “pydicom” enables loading of DICOM files into Python objects for easy access

MacLennan 2018 76
MCI: Bypassing the TPS

- Treatment Planning System
- Scripting API
- Scripting System

MCI: What’s Inside DICOM

- DICOM is simultaneously a complex format and a simple format
  - Each piece is simple
  - Simple pieces have to be put in just the right spot
MCI: Simplified RT-Struct

- DICOM **RT-Struct** stores ROIs

- Most of an RT-Struct file is just long lists of Connect-the-Dot coordinates

MCI: Stray Voxel Finder

- To find disconnected stray voxels, look through all the lists of contours and find any with fewer than 5 points

```
for each ROI in rtstruct.ROIContourSequence
    for each contour_slice in ROI.ContourSequence
        if contour_slice.NumberOfContourPoints < 5
            print an alert
```
• DICOM plan files are more complex
• We can selectively examine some branches of the whole tree

plan
   IonBeamSequence[]
   BeamName
   IonControlPointSequence[]
      GantryAngle
      PatientSupportAngle

• To access the gantry angle:
  plan.IonBeamSequence.IonControlPointSequence[0].GantryAngle

• To access the beam name:
  plan.IonBeamSequence.BeamName

• Does the text of the gantry angle show up inside the beam name?
Does this appear in there?

Looks like that comparison...
...has met its match

MCI: Miami

YEAAAAAAAAAAAA
Quality Checks

• So far we’ve focused on black/white concepts like detecting errors
• Scripts can also focus on gray-area ideas like plan quality
  – Does a plan meet clinical goals?
    • Target coverage
    • OAR sparing
    • Conformality
  – If not, how close were we?
**SBRT Worksheets**

- Currently at MCI, SBRT cases are evaluated by manually filling out an Excel spreadsheet.

**PTV Parameters**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>PTV Volume [cm^3]</td>
<td>209.18</td>
</tr>
<tr>
<td>Vol of PTV receiving Pres. Dose [cm^3]</td>
<td>375.37</td>
</tr>
<tr>
<td>100% Pres. IsoVold Vol. [cm^3]</td>
<td>188.57</td>
</tr>
<tr>
<td>95% Pres. IsoVold Vol. [cm^3]</td>
<td>633.76</td>
</tr>
<tr>
<td>Max Dose at 2.0cm reg [Gy]</td>
<td>21.9</td>
</tr>
</tbody>
</table>

- OARs must be chosen based on treatment site and goal values must be entered manually.

**Normal Tissue Parameters**

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Spinal Cord</td>
<td>0.4</td>
<td>22.0</td>
<td>28.0</td>
<td>Yes</td>
</tr>
<tr>
<td></td>
<td>1.2</td>
<td>15.3</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Esophagus</td>
<td>5.0</td>
<td>27.5</td>
<td>35.0</td>
<td>Yes</td>
</tr>
<tr>
<td>Lung R Only</td>
<td>37%</td>
<td>17.5</td>
<td>18.5</td>
<td>Yes</td>
</tr>
<tr>
<td></td>
<td>750.0</td>
<td>12.5</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Skin</td>
<td></td>
<td></td>
<td>38.5</td>
<td></td>
</tr>
<tr>
<td>Heart</td>
<td>15.0</td>
<td>32.0</td>
<td>38.0</td>
<td></td>
</tr>
</tbody>
</table>
Clinical Goal Lookup

- Single fraction
- Three fraction
- Five fraction

SBRT Worksheets

- DVH values must be manually looked up and written into the spreadsheet
SBRT Worksheets

- DVH values must be manually looked up and written into the spreadsheet

<table>
<thead>
<tr>
<th>Normal Tissue Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Critical OAR</strong></td>
</tr>
<tr>
<td>-------------------</td>
</tr>
<tr>
<td>Spinal Cord</td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td>Esophagus</td>
</tr>
<tr>
<td>Lung Only</td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td>Skin</td>
</tr>
<tr>
<td>Heart</td>
</tr>
</tbody>
</table>
**SBRT Worksheets**

- Clinical goal pass/fail must still be determined manually and recorded.

### Normal Tissue Parameters

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Spinal Cord</td>
<td>0.4</td>
<td>12.8</td>
<td>14.4</td>
<td>Yes</td>
</tr>
<tr>
<td></td>
<td></td>
<td>22.0</td>
<td>28.0</td>
<td></td>
</tr>
<tr>
<td>Esophagus</td>
<td>5.0</td>
<td>24.3</td>
<td>29.3</td>
<td>Yes</td>
</tr>
<tr>
<td>Lung R Only</td>
<td>37%</td>
<td>8.0</td>
<td>13.5</td>
<td>Yes</td>
</tr>
<tr>
<td></td>
<td></td>
<td>750.0</td>
<td>7.2</td>
<td>Yes</td>
</tr>
<tr>
<td></td>
<td></td>
<td>12.5</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Skin</td>
<td></td>
<td></td>
<td>25.5</td>
<td>Yes</td>
</tr>
<tr>
<td>Heart</td>
<td>15.0</td>
<td>9.3</td>
<td>12.2</td>
<td>Yes</td>
</tr>
<tr>
<td></td>
<td></td>
<td>32.0</td>
<td>38.0</td>
<td></td>
</tr>
</tbody>
</table>

**MacLennan 2018**

### SBRT Worksheets

- Oops! To save space, different goals were crammed onto the same line on this sheet.

### Normal Tissue Parameters

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Spinal Cord</td>
<td>0.4</td>
<td>12.8</td>
<td>14.4</td>
<td>Yes</td>
</tr>
<tr>
<td></td>
<td></td>
<td>22.0</td>
<td>28.0</td>
<td></td>
</tr>
<tr>
<td>Esophagus</td>
<td>5.0</td>
<td>24.3</td>
<td>29.3</td>
<td>Yes</td>
</tr>
<tr>
<td>Lung R Only</td>
<td>37%</td>
<td>8.0</td>
<td>13.5</td>
<td>Yes</td>
</tr>
<tr>
<td></td>
<td></td>
<td>750.0</td>
<td>7.2</td>
<td>Yes</td>
</tr>
<tr>
<td></td>
<td></td>
<td>12.5</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Skin</td>
<td></td>
<td></td>
<td>25.5</td>
<td>Yes</td>
</tr>
<tr>
<td>Heart</td>
<td>15.0</td>
<td>9.3</td>
<td>12.2</td>
<td>Yes</td>
</tr>
<tr>
<td></td>
<td></td>
<td>32.0</td>
<td>38.0</td>
<td></td>
</tr>
</tbody>
</table>

**MacLennan 2018**
Is There a Better Way?

- Tedious manual processes are error-prone

Automating Clinical Goals

- Several commercial treatment planning systems already incorporate some level of automated clinical goal checking.
  - Some can load pre-made templates
  - Some require you to set up goals by hand
Commercial Software

- There is plenty of 3rd party plan check software available for purchase too
- Does not require any programming
- DOES require some setup and configuration

Example: ClearCheck

- Uses the Eclipse Scripting API (ESAPI)
- Runs as a Plugin inside Eclipse
  - The actual interface appears as a new window
- ClearCheck runs a wide variety of tests
  - Plan parameters and ROI checks
  - Clinical goals scorecard
Clinical Goals in Eclipse

- ClearCheck’s clinical goals scorecard is especially appealing for Eclipse
- Eclipse has a protocols feature that can load goal templates
  - Specific to protocol, not patient
- ClearCheck can load an initial template and then customize it for each patient
Don’t Need Spreadsheet

- All of the data that had to be manually entered into the Excel spreadsheet is automatically pulled from the plan
- Could potentially work on imported plans from other treatment planning systems too

Time to Wrap Up
Some Takeaways

• Automated plan checks are a hot topic
  – Multiple home-brew solutions
  – Multiple commercial solutions

• Safety is enhanced and quality can get more attention when automation helps out

• You can do interesting projects with only basic understanding of coding
  – The more you learn, the fancier it can be

Programming Secret

• Most of programming is just getting good at using Google
Acknowledgements

- Fazal Khan, CMD
  - Miami Cancer Institute, Miami, FL
- Craig McKenzie, CMD
  - Miami Cancer Institute, Miami, FL
- Samantha Hedrick, PhD, DABR
  - Emory Proton Therapy Center, Atlanta, GA
- Eric Ford, PhD, FAAPM
  - University of Washington Medical Center, Seattle, WA
- Patricia Sponseller, MS, CMD, RTT(R)(T)
  - University of Washington Medical Center, Seattle, WA
- Hsiao-Ming Lu, PhD
  - Massachusetts General Hospital, Boston, MA