VARIAN
medical systems

Eclipse Scripting API Reference Guide for Research Users

Eclipse

P1008615-002-B

OCTOBER 2014

Document ID P1008615-002-B

Document Title *Eclipse Scripting API Reference Guide for Research Users*

Abstract This document provides information about using the research features in Eclipse Scripting API, version 13.6. This publication is the English-language original.

Manufacturer

Varian Medical Systems, Inc.
3100 Hansen Way
Palo Alto, CA 94304-1038
United States of America

European Authorized Representative

Varian Medical Systems UK Ltd.
Oncology House
Gatwick Road, Crawley
West Sussex RH10 9RG
United Kingdom

FDA 21 CFR 820 Quality System Regulations (cGMPs) Varian Medical Systems, Oncology Systems products are designed and manufactured in accordance with the requirements specified within this federal regulation.

ISO 13485 Varian Medical Systems, Oncology Systems products are designed and manufactured in accordance with the requirements specified within the ISO 13485 quality standard.

CE Varian Medical Systems, Oncology Systems products meet the requirements of Council Directive MDD 93/42/EEC.

IEC62083 Eclipse™ Treatment Planning System is IEC62083:2009 compliant.

EU REACH SVHC Disclosure The link to the current EU REACH SVHC disclosure statement can be found at http://www.varian.com/us/corporate/legal/reach.html.

HIPAA Varian's products and services are specifically designed to include features that help our customers comply with the Health Insurance Portability and Accountability Act of 1996 (HIPAA). The software application uses a secure login process, requiring a user name and password, that supports role-based access. Users are assigned to groups, each with certain access rights, which may include the ability to edit and add data or may limit access to data. When a user adds or modifies data within the database, a record is made that includes which data were changed, the user ID, and the date and time the changes were made. This establishes an audit trail that can be examined by authorized system administrators.

WHO

ICD-0 codes and terms used by permission of WHO, from:

- International Classification of Diseases for Oncology, (ICD-0) 3rd edition, Geneva, World Health Organization, 2000.

ICD-10 codes and terms used by permission of WHO, from:

- International Statistical Classification of Diseases and Related Health Problems, Tenth Revision (ICD-10). Vols 1-3, Geneva, World Health Organization, 1992.

CAUTION: US Federal law restricts this device to sale by or on the order of a physician.

Contents

Chapter 1 Introduction

Eclipse is used to plan radiotherapy treatments for patients with malignant or benign diseases. The users of Eclipse are medical professionals who have been trained in radiation dosimetry. After an oncologist has decided that radiotherapy is the suitable treatment for a patient, the medical professionals use Eclipse to plan the treatment for the patient. Eclipse can be used to plan external beam irradiation with photon, electron, and proton beams, as well as for internal irradiation (brachytherapy) treatments. Eclipse is part of Varian's integrated oncology environment.

The Eclipse Scripting Application Programming Interface (Eclipse Scripting API or ESAPI) is a programming interface and a software library for Eclipse. It allows software developers to write scripts to access the treatment planning information in Eclipse. The scripts can be integrated into the Eclipse user interface, or they can be run as stand-alone executables.

The Eclipse Scripting API can be licensed for non-clinical research and development use by Varian's research partners. When licensed in this manner, plan and structure data can be modified and saved to a database that is dedicated to non-clinical research use. This manual describes the Eclipse Scripting API that is available when licensed for research.

Who Should Read This Manual

This manual is written mainly for medical/technical personnel who wish to write custom scripts for modifying plan and structure data in a research database. It is assumed that you are familiar with:

- Eclipse Treatment Planning System
- Radiation oncology domain and concepts
- DICOM
- Software engineering practices
- Microsoft Visual Studio development environment
- Microsoft Visual C# programming language and object oriented development

Note *Before creating your own scripts, familiarize yourself with the Eclipse user documentation, especially any safety-related information, cautions, and warnings found throughout the documentation.*

Visual Cues

This publication uses the following visual cues to help you find information:

WARNING: **A warning describes actions or conditions that can result in serious injury or death.**

CAUTION: **A caution describes hazardous actions or conditions that can result in minor or moderate injury.**

NOTICE **A notice describes actions or conditions that can result in damage to equipment or loss of data.**

Note *A note describes information that may pertain to only some conditions, readers, or sites.*

Tip *A tip describes useful but optional information such as a shortcut, reminder, or suggestion, to help get optimal performance from the equipment or software.*

Related Publications

- RT Administration Reference Guide P1008617-001-A
- Beam Configuration Reference Guide P1008610-001-A
- BrachyVision Instructions for Use P1005650-001
- BrachyVision Reference Guide P1005651-001
- Acuros BV Algorithm Reference Guide B504878R01A
- Eclipse Photon and Electron Algorithms Reference Guide P1008611-001-A
- Eclipse Cone Planning Online Help B504830R01A
- Eclipse Ocular Proton Planning Reference Guide P1005367-001-A
- Eclipse Photon and Electron Instructions for Use P1008620-001-A
- Eclipse Photon and Electron Reference Guide P1008621-001-A
- Eclipse Proton Reference Guide P1008623-001-A
- Eclipse Proton Instructions for Use P1008622-001-A
- Eclipse Proton Algorithms Reference Guide B504886R01A
- Eclipse Scripting API Online Help for Research Users P1008613-001-A
- Eclipse Scripting API Reference Guide P1008614-002-B

Contacting Varian Customer Support

Varian Customer Support is available on the internet, by e-mail, and by telephone. Support services are available without charge during the initial warranty period.

The my.varian.com website provides contact information, product documentation, and other resources for all Varian products.

Get Online Customer Support

You can browse the my.varian.com site without having a Varian account or logging in. However, you must have a Varian account to get online customer support and to access product information for products at your institution or clinic.

1. Go to http://my.varian.com.
2. Click **Contact Us** at the top of the window to display customer support and training options, and international e-mail addresses and telephone numbers.
3. Choose an option:
   - If you do not already have an account, click **Create New Account** and follow the instructions. Establishing an account may take a few days.
   - If you have an account, go to the next step.
4. Enter your user name and password.
5. Browse the information and then click the link that corresponds to what you want to do:
   - Fill out and submit a support request.
   - Find documents. Online documents in PDF format include customer technical bulletins (CTBs,) manuals, and customer release notes (CRNs).
   - Send an e-mail to Varian support. You can browse for international e-mail addresses and telephone numbers by geographic area, and for oncology-specific contacts such as for brachytherapy.
   - Find parts and services by geographical area.

E-Mailing Varian

Send e-mail inquiries through the my.varian.com website.

Alternatively, you can use a support e-mail address that corresponds to your location or interest:

Location	E-mail Address
North America	support-americas@varian.com
Latin America	soporte.al@varian.com
Europe	support-emea@varian.com
Australia and New Zealand	support-anz@varian.com
China	support-china@varian.com

Japan	support-japan@varian.com
South East Asia	support-sea@varian.com
Brachytherapy Systems	brachyhelp@varian.com

Ordering Documents by Phone

You can order documents by phone by calling Varian Medical Systems support.

Location	Telephone Number
North America	+ 1 888 827 4265 (Press 2 for parts)
Global	Call your local Varian office.

Chapter 2 Research Features in the Eclipse Scripting API

The Eclipse Scripting API is a Microsoft .NET class library that gives you access to the treatment planning data of Eclipse. It allows you to create scripts that leverage the functionality of Eclipse, and lets you retrieve plan, image, dose, structure, and DVH information from the Varian System database. The data is retrieved from the Varian System database also in stand-alone Eclipse installations. You can integrate the scripts into Eclipse, or you can run them as stand-alone executables.

Additional API features are available for non-clinical research and development use when the system is configured as a research environment. A research environment has an Eclipse Scripting API research license installed, and the Varian System database has been configured for research use. The configuration is done by the Varian service personnel at the time of installation.

Research environments are different from clinical environments in that:

- Users cannot treatment-approve plans, which prevents treating from a research environment database.
- The title bars in Eclipse workspaces note that the system is running in a research environment.
- All Eclipse printouts generated from this environment note that they were created in a research environment.
- The Beam Configuration application warns that changes should only be made if the DCF environment is dedicated for research.
- All plans created or modified through the Eclipse Scripting API have their intent set to “Research”.

In research environments, users are given access to all Eclipse Scripting API features available in clinical environments. They are also given access to scripting features that allow for creation and modification of structure and plan data, and execution of dose calculation and optimization algorithms. The research features are available for photon plans.

WARNING: **The authors of custom scripts are responsible for verifying the accuracy and correctness of the scripts after developing a new script or after system upgrade for the existing scripts.**

Features

By using the research features in the Eclipse Scripting API, you can create scripts to:

- Create and modify structures.
- Create and modify plans and fields.
- Create artificial phantom images.
- Optimize plans by using the Eclipse optimization algorithms.
- Calculate leaf motions by using the Eclipse leaf motion calculation algorithms.
- Calculate dose by using the Eclipse dose calculation algorithms.
- Execute DVH Estimation.

System Requirements

The basic system requirements of the research features in the Eclipse Scripting API are the same as those of Eclipse. For more information, refer to *Eclipse Customer Release Note*.

To create write-enabled research scripts with the Eclipse Scripting API 13.6, you need the following:

- Eclipse 13.6 or later.
- A license for the Eclipse Scripting API 13.6.
- A license that enables research features in the Eclipse Scripting API.
- A non-clinical Varian System database configured for research use.

Varian Medical Systems provides no guarantee that scripts written with this version of the Eclipse Scripting API will be compatible with future releases.

For more information about the Eclipse Scripting API, refer to *Eclipse Scripting API Reference Guide*.

Version Compatibility

ESAPI 13.6

- The Research Features in the Eclipse Scripting API 13.6 are compatible with Eclipse 13.6.
- Varian Medical Systems provides no guarantee that scripts written with this version of the Eclipse Scripting API will be compatible with future releases.

Incompatibilities between ESAPI 13.5 and 13.6

- Method `VMS.TPS.Common.Model.API.Dose.SetVoxels` has been marked as obsolete. Functionality is now provided in `VMS.TPS.Common.Model.API.EvaluationDose.SetVoxels`.

ESAPI 13.5

- The Research Features in the Eclipse Scripting API 13.5 are compatible with Eclipse 13.5.
- Varian Medical Systems provides no guarantee that scripts written with this version of the Eclipse Scripting API will be compatible with future releases.

Incompatibilities between ESAPI 13.0 and ESAPI 13.5

- Method `VMS.TPS.Common.Model.OptimizationSetup.AddStructurePointCloudParameter` has been marked as obsolete. Point cloud resolution is no longer an input to optimization.

Note *If you use an obsoleted type, property, field, or method, the compiler shows a warning. In this case, the compilation of a single-file plug-in fails. If the script is a binary plug-in or a standalone executable, the compiler shows an error. This happens only if the "Treat warnings as errors" project setting is turned on in Microsoft Visual Studio.*

Upgrade ESAPI 13.5 to ESAPI 13.6

For instructions on upgrading from a previous version of ESAPI, refer to *Eclipse Scripting API Reference Guide*.

What Is New in Eclipse Scripting API for Research Users 13.6

In this version of the Eclipse Scripting API for Research Users you can do the following:

- Create scripts that use intermediate dose calculation during IMRT and VMAT optimization. See the new methods `VMS.TPS.Common.Model.API.ExternalPlanSetup.Optimize(OptimizationOptionsIMRT)` and `VMS.TPS.Common.Model.API.ExternalPlanSetup.OptimizeVMAT(OptimizationOptionsVMAT)`.
- Terminate IMRT optimization automatically upon convergence instead of using the specified number of iterations. See the new method `VMS.TPS.Common.Model.API.ExternalPlanSetup.Optimize(OptimizationOptionsIMRT)`.
- Copy a 3D image from another patient. See the new method `VMS.TPS.Common.Model.API.Patient.CopyImageFromOtherPatient`.
- Add an external beam plan as a verification plan. See the new method `VMS.TPS.Common.Model.API.Course.AddExternalPlanSetupAsVerificationPlan`.
- Set a three-dimensional asymmetric margin around a structure. See the new method `VMS.TPS.Common.Model.API.SegmentVolume.AsymmetricMargin`.

The following new properties, functions, and classes have been added or changed:

- A new class `VMS.TPS.Common.Model.API.EvaluationDose` has been added and `SetVoxels` functionality has been moved there from the base class `VMS.TPS.Common.Model.API.Dose`.
- `VMS.TPS.Common.Model.API.ExternalPlanSetup`: added a property `DoseAsEvaluationDose`.

For what is new in Eclipse Scripting API 13.6, refer to *Eclipse Scripting API Reference Guide.*

Chapter 3 Creating and Saving Write-enabled Scripts

You can create and save write-enabled scripts as described below.

Creating Write-Enabled Scripts

To create a write-enabled script, use the Eclipse Script Wizard as instructed in *Eclipse Scripting API Reference Guide*. Then add a call to the `BeginModifications` method of the `Patient` class as shown in the code example below:

```
namespace VMS.TPS
{
  class Script
  {
    public Script()
    {
    }

    public void Execute(ScriptContext context, System.Windows.Window window)
    {
      Patient patient = context.Patient;
      // BeginModifications will throw an exception if the system is not
      // configured for research use.
      patient.BeginModifications();

      // After calling BeginModifications successfully it is possible
      // to modifiy patient data.
      if (patient.CanAddCourse())
      {
        // E.g. the script adds a new course
        Course newCourse = patient.AddCourse();
        // Continue with other changes
      }
    }
  }
}
```

Saving Modifications in Stand-alone Executable Scripts

In a stand-alone executable script, use the `Application` class to save or discard modifications:

- To save the modifications to the database, call the `SaveModifications` method.
- To discard the modifications, close the patient by calling the `ClosePatient` method. You can then open the patient again if the script still needs to access the patient data.

Saving Modifications in Plug-in Scripts

A plug-in script (single-file or binary) modifies the current data context of the Eclipse Treatment Planning application.

After the script has been executed, you can save or discard the modifications in the Eclipse Treatment Planning user interface in the same way as any other change.

Chapter 4 Using the Research Features

You can use the research features in the Eclipse Scripting API for the tasks listed below.

Adding and Modifying Structures

You can add, remove, and modify structures by using the classes described below.

Adding and Removing Structures

Use the `StructureSet` class to add and remove structures:

- Add structures with the `AddStructure` method. You can use the `CanAddStructure` method to check if the script is able to add a new structure to the structure set.

  To specify the type of structure to add, give the string representation of the DICOM type as a parameter to the `AddStructure` method. Example values are `EXTERNAL`, `ORGAN`, and `PTV`. For a complete list of allowed values, refer to *Eclipse Scripting API Online Help for Research Users*.

- Remove structures with the `RemoveStructure` method.

Modifying Structures

Use the `Structure` class to modify structures:

- Add contours to a structure with the `AddContourOnImagePlane` method.

  Input parameters: a list of points defining the contour, and the index of the image plane where the contours are to be added.

- Subtract contours with the `SubtractContourOnImagePlane` method.

  Input parameters: a list of points defining the contour, and the index of the image plane where the contours are to be subtracted.

- Clear all contours for a structure with the `ClearAllContoursOnImagePlane` method.

- Set a structure to the result of a Boolean operation with the `And`, `Or`, `Not`, and `Xor` methods. These methods are available also on the `SegmentVolume` class, which allows you to execute a combination of Boolean operations with an intermediate variable of the `SegmentVolume` type before assigning the final result to a `Structure` object. For more information, see the `SegmentVolume` property getter and setter of the `Structure` class.

- Set a three-dimensional symmetric or asymmetric margin around a structure. First call the `Margin` or `AsymmetricMargin` method. Then set the resulting `SegmentVolume` object to the `Structure` object using the `SegmentVolume` property. For more information, see the get and set accessors of the `SegmentVolume` property of the `Structure` class.

Adding and Removing Artificial Phantom Images

Use the `Patient` class to add and remove artificial phantom images:

- Add a phantom image with the `AddEmptyPhantom` method.

  Input parameters: patient orientation, the size of the image set in X- and Y-directions in pixels and in millimeters, the number of planes, and the separation between the planes in millimeters.

  The image is created to a new `Study`. The return value of the `AddEmptyPhantom` is a new `StructureSet` where you can add structures and their contours. You can access the created `Image` using the `Image` property of the `StructureSet` class.

- Remove a phantom image and associated `StructureSet` with the `RemoveEmptyPhantom` method.

  Input parameter: `StructureSet`. The `StructureSet` must not contain any `Structures`. Before removing the phantom, you can use the `CanRemoveEmptyPhantom` method to check if the script is able to remove the `Image`.

Copying an Image from Another Patient

Use the `Patient` class to copy a 3D image from another patient:

- Specify the identifiers of the other patient, the study of the other patient, and the 3D image of the other patient. Validate the identifiers using the method `CanCopyImageFromOtherPatient`. Copy the image using the method `CopyImageFromOtherPatient`.

  Input parameters: the identifier of the other patient, the identifier of the study of the other patient, the identifier of the 3D image of the other patient.

Creating and Modifying Plans and Fields

You can create and modify plans and fields by using the classes described below.

Adding and Removing Plans

Use the `Course` class to add and remove plans:

- Add a new external beam photon plan with the `AddExternalPlanSetup` method.

  Input parameter: `StructureSet`. A new primary reference point without a location is automatically created for the plan. You can use the `CanAddPlanSetup` method to check if the script is able to add a new plan to the course.

- Add a new external beam photon plan as a verification plan with the `AddExternalPlanSetupAsVerificationPlan` method.

  Input parameters: structure set, the verified plan.

- Remove a plan with the `RemovePlanSetup` method.

Input parameter: `PlanSetup`. You can use `CanRemovePlanSetup` to check if the script is able to remove the plan from the course.

Adding Fields

Use the `ExternalPlanSetup` class to add fields:

- Add a new static open field to the plan with the `AddStaticBeam` method.
- Add a new arc field to the plan with the `AddArcBeam` method.
- Add a field with static MLC shape with the `AddMLCBeam` or `AddMLCArcBeam` method.
- Add an arc field with dynamic MLC shape using the `AddConformalArcBeam` method.
- Add an IMRT field with the Multiple Static Segment delivery method using the `AddMultipleStaticSegmentBeam` method.
- Add an IMRT field with the Sliding Windows delivery method using the `AddSlidingWindowBeam` method.
- Add a VMAT field with the `AddVMATBeam` method.

Define the `TreatmentMachine` configuration for added fields:

- Create an `ExternalBeamMachineParameters` object and use it as input for the above mentioned beam adding methods of the `ExternalPlanSetup` class.
- Input parameters for the `ExternalBeamMachineParameters` object: the Treatment Machine identifier, Energy Mode identifier, Field Technique identifier, Dose Rate, and optionally, the Primary Fluence Mode identifier.

For information about other input parameters for the beam adding methods of the `ExternalPlanSetup` class, see *Eclipse Scripting API Online Help for Research Users*.

Modifying Fields

Use the `BeamParameters` class to modify a `Beam`:

- Call the `GetEditableParameters` method. Modify the returned `BeamParameters` object. Call `ApplyParameters`. If the call succeeds, the beam data is updated.
- Set the optimal fluence of an IMRT field with the `SetOptimalFluence` method.
- Modify `ControlPoints` using the `ControlPoints` property of the `BeamParameters` class. Each `ControlPoint` has one `ControlPointParameters` object. Call `ApplyParameters`. If the call succeeds, the beam data is updated.

Adding Prescriptions and Fractionation

- To access the `Fractionation` object of a plan, use the `UniqueFractionation` property of the `PlanSetup` object.

- Set the prescription with the `SetPrescription` method of the `Fractionation` object.

  Input parameters for the method: the number of fractions, the prescribed dose per fraction (using the dose unit defined for your system in the Varian Oncology System Platform portal), and prescribed percentage.

Using Calculation Algorithms

You can use the classes described below for IMRT and VMAT optimization, leaf motion calculation, and dose calculation.

Setting Calculation Models

Use the `PlanSetup` class to set the calculation models:

- Set the calculation model with the `SetCalculationModel` method.

  Input parameters: calculation type and calculation model name.

  The current calculation model can be retrieved with the `GetCalculationModel` method.

  All available models can be read with the `GetModelsForCalculationType` method of the `ExternalPlanSetup` class.
- Set the calculation options using the `SetCalculationOption` method. The available options and their allowed values depend on the calculation model. Before setting the calculation options, set the calculation model.
- Clear the calculation model of a plan with the `ClearCalculationModel` method.

Viewing Calculation Logs

Use the following classes to read the messages from the algorithms:

- `Beam`

  Access the calculation logs after calculation with the `CalculationLogs` property. It returns a collection of `BeamCalculationLog` objects that can be filtered using the `Category` property. The category for the optimization log is, for example Optimization.
- `System.Diagnostics.Trace`

  Use the Trace class of the .NET framework to receive messages from the algorithms during the calculation. For more information about how to use Trace Listeners, consult the Microsoft Developer Network (MSDN) documentation.

Executing DVH Estimation

Use the `ExternalPlanSetup` class for executing DVH Estimation.

- Run DVH Estimation with the `CalculateDVHEstimates` method.

Input parameters: the identifier of the DVH Estimation model, the dose level for target structure(s) and the mapping between the structures of the estimation model and the structureset of the plan. Use the `Success` property of `OptimizerResult` to check if the algorithm executed without errors.

Optimizing IMRT and VMAT Plans

Use the following classes for optimizing IMRT and VMAT plans.

Setting up the Plan

- Before starting the optimization, create open fields for the plan with the `AddStaticBeam` or `AddArcBeam` method of the `ExternalPlanSetup` class.
- Use the `OptimizationSetup` property of the `PlanSetup` class to access the `OptimizationSetup` object.

Adding and Modifying Optimization Objectives

Use the `OptimizationSetup` object to add and modify optimization objectives:

- Add point objectives with the `AddPointObjective` method.
- Add mean dose or gEUD objectives with the `AddMeanDoseObjective` or `AddEUDObjective` methods.
- Add beam-specific parameters with the `AddBeamSpecificParameter` method.
- Add a Normal Tissue Objective with the `AddNormalTissueObjective` method.
- Remove optimization objectives with the `RemoveObjective` method.
- Remove optimization parameters with the `RemoveParameter` method.

Optimizing an IMRT Plan

Use the `ExternalPlanSetup` class to optimize an IMRT plan:

- Run the IMRT optimization algorithm with the `Optimize` method.

  Input parameter: the number of needed iterations. All existing optimal fluences are removed. As a result of `Optimize`, the `OptimizerResult` object is returned. Use the `Success` property of `OptimizerResult` to check if the algorithm executed without errors.
- You can continue IMRT optimization with the overloaded version of the `Optimize` method, for which you can give the `OptimizationOption` as a parameter. To continue optimization with existing optimal fluences, use the `OptimizationOption.ContinueOptimization` as a parameter.
- You can continue IMRT optimization with the existing plan dose as an intermediate dose. To do this, use the overloaded version of the `Optimize` method, and define the `OptimizationOption.ContinueOptimizationWithPlanDoseAsIntermediateDose` as a parameter.

- You can terminate IMRT optimization upon convergence. To do this, use the `Optimize` method that does not take any input parameters, or the overloaded version of the `Optimize` method that takes an `OptimizationOptionsIMRT` object as an input. Construct the `OptimizationOptionsIMRT` object using the `OptimizationConvergenceOption.TerminateIfConverged` option. After optimization you can read the actual number of iterations from `OptimizerResult.NumberOfIMRTOptimizerIterations`.
- You can use intermediate dose calculation during IMRT optimization. To do this, create an `OptimizationOptionsIMRT` object and specify the `numberOfStepsBeforeIntermediateDose`. Then use the overloaded version of `Optimize` method that takes an `OptimizationOptionsIMRT` object as an input.

Optimizing a VMAT Plan

Use the `ExternalPlanSetup` class to optimize a VMAT plan:

- Run the VMAT optimization algorithm with the `OptimizeVMAT` method. As a result of `Optimize`, the `OptimizerResult` object is returned. Use the `Success` property of `OptimizerResult` to check if the algorithm executed without errors.
- You can use intermediate dose calculation during VMAT optimization. To do this, create an `OptimizationOptionsVMAT` object by specifying either the option `OptimizationIntermediateDoseOption.UseIntermediateDose` or the number of optimization cycles. Then use the overloaded version of method `OptimizeVMAT` that takes an `OptimizationOptionsVMAT` object as an input parameter.

Accessing Dose Volume Histograms

Use the `OptimizerResult` class to access the results of the optimization:

- Use the `StructureDVHs` property to access the Dose Volume Histograms after optimization. The returned collection contains an `OptimizerDVH` object for each `Structure` that had optimization objectives defined. Use the `CurveData` property of the `OptimizerDVH` class to access the points of the Dose Volume Histogram.

Accessing Optimal Fluences after IMRT Optimization

Use the `Beam` class to access the optimal fluence information:

- Read the optimal fluence matrix after optimization with the `GetOptimalFluence` method.

Calculating Leaf Motions after IMRT Optimization

Use the `ExternalPlanSetup` class to execute the leaf motion calculation algorithm:

- Calculate leaf motions with the `CalculateLeafMotions` method. The method uses the default calculation options of the leaf motion calculation model set in the plan. The method returns a `CalculationResult` object. This object has a `Success` property, which you can use to check if the algorithm executed without errors.

- To run the Varian Leaf Motion Calculator algorithm, use the overloaded `CalculateLeafMotions` method.

  Input parameter: an `LMCVOptions` object. In `LMCVOptions`, you can specify the usage of fixed jaws. Check that the leaf motion calculation model of the plan is Varian Leaf Motion Calculator.

- To run the Varian Smart LMC algorithm, use the overloaded `CalculateLeafMotions` method.

  Input parameter: a `SmartLMCOptions` object. In `SmartLMCOptions`, you can specify the usage of fixed field borders and jaw tracking. Check that the leaf motion calculation model of the plan is Varian Smart LMC.

- To run the non-Varian MSS Leaf Motion Calculator algorithm, use the overloaded `CalculateLeafMotions` method.

  Input parameter: an `LMCMSSOptions` object. In `LMCMSSOptions`, you can specify the number of calculation iterations. Check that the leaf motion calculation model of the plan is MSS Leaf Motion Calculator.

Calculating Dose

Use the `ExternalPlanSetup` class for dose calculation:

- Calculate the volume dose using the `CalculateDose` method. The method returns a `CalculationResult` object. This objecs has a `Success` property that you can use to check if the algorithm executed without errors.
- Calculate the volume dose with preset MUs using the `CalculateDoseWithPresetValues` method.

  Input parameter: a list of `Beam` identifier and `MeterSetValue` pairs of the `Beam`.