```
In[@]:= QMRITools`$Verbose = False;
       << QMRIToolsDev`
 In[@]:= Grid[Transpose@{{"Packages", "Functions", "Options"}, Length@Flatten[QMRIToolsFunctions[][All, #]] & /@ {1, 2, 3}}, Alignment → Left]
Out[0]=
      Packages 25
      Functions 491
      Options 435
 In[@]:= Column@QMRIToolsPackages[]
Out[0]=
      CardiacTools
      CoilTools
      DenoiseTools
      DixonTools
      ElastixTools
      FasciculationTools
      GeneralTools
      GradientTools
      IVIMTools
      JcouplingTools
      LoggingTools
      MaskingTools
      MuscleBidsTools
      NiftiTools
      PlottingTools
      ProcessingTools
      ReconstructionTools
      RelaxometryTools
      ScientificColorData
      SegmentationTools
      SimulationTools
      SpectroTools
      TaggingTools
      TensorTools
      TractographyTools
 In[@]:= QMRIToolsFunctions["All", 4]
```

QMRITools

2 All-Functions.nb ADCCalc	ExportBval	Link3DGraphic	RegisterCardiacData
AddLossLayer	ExportBvec	ListSpherePlot	RegisterData
AddNoise	ExportDirectoryTree	LLeastSquares	RegisterDataSplit
AddScientificColors	ExportLog	LoadCoilSetup	RegisterDataTransform
AddToJson	ExportNii	LoadCoilTarget	RegisterDataTransformSplit
AddToLog	ExportSparSdat	LogNoZero	RegisterDiffusionData
AnalyseNetworkFeatures	ExportTracts	_	RegisterDiffusionDataSplit
=	•	LogTensor MADNoZero	
AnalyzeActivations	ExpTensor ExtractColorData	MakeChannelClassGrid	RegisterTensorData
AngleCalc			RemoveIsoImages
AngleMap	ExtractDemoData	MakeChannelClassImage	RemoveMaskOverlaps
AnisoFilterData	ExtractFromJSON	MakeChannelImage	ReplaceSegmentations
AnisoFilterTensor	ExtractNiiFiles	MakeCheckFile	ResampleTracts
AnnalyzeTagging	FACalc	MakeClassifyImage	RescaleData
ApodizeEcho	FConvert	MakeClassifyNetwork	RescaleSegmentation
ApodizeFid	FConverti	MakeClassImage	RescaleTracts
ApodizePadEcho	FiberLength	MakeCoilLayout	ResetLog
ApodizePadFid	FiberTractography	MakeCoordinates	ResidualCalc
ApodizePadSpectra	FileSelect	MakeDistanceMap	ReverseCrop
ApodizeSpectra	FilterTracts	MakeECVBloodMask	ReverseDimensions
ApplyCrop	FinalGrads	MakeFunctionGraph	RMSNoZero
ApplySegmentationNetwork	FindActivations	MakeHammingFilter	RotateData
AugmentImageData	FindCoilPosition	MakeIntFunction	RotateDimensionsLeft
AugmentTrainingData	FindCrop	MakeLineImage	RotateDimensionsRight
AutoCropData	FindInPhaseEchos	MakeMaskImage	RotateTensor
B1MapCalc	FindMaxDimensions	MakeNiiOrentationQ	RotationMatrixToQuaternion
B1Shimming	FindOrder	MakeNiiOrentationS	RotationMatrixToQuaternionVector
BayesianIVIMFit2	FindOutliers	MakeNode	SagitalTranspose
BayesianIVIMFit3	FindSpectraPpmShift	MakeNoisePlots	SaveImage
BidsDcmToNii	FindTensorPermutation	MakeSense	SeedDensityMap
BlochSeries	FitData	MakeSliceImages	SegmentationVolume
Bmatrix	FitGradientMap	MakeSpectraResultPlot	SegmentData
BmatrixCalc	FitSpectra	MakeSpinSystem	SegmentDataGUI
BmatrixConv	FitSpectraResultTable	MakeUnet	SegmentLinesToMask
BmatrixInv	FitTracts	MakeWeightMask	SegmentMask
BmatrixRot	FixDixonFlips	Mask	SegmentsPerSlice
BmatrixToggle	FlipGradientOrientation	MaskData	SegmentTracts
BSplineCurveFit	FlipTensorOrientation	MaskHelix	SelectActivations
BullseyePlot	FocalLossLayer	MaskSegmentation	SelectBidsFolders
CalculateDispacementParameters	FourierKspace2D	MaskToLines	SelectBidsSessions
CalculateGfactor	FourierKspace3D	MBCount	SelectBidsSubjects
CalculateMoments	FourierKspaceCSI	MeanBvalueSignal	SelectBvalueData
CalculateWallMap	FourierRescaleData	MeanNoZero	SelectMaskComponents
CalibrateEPGT2Fit	FourierShift	MeanRange	SelectSegmentations
CardiacCoordinateSystem	FourierShifted	MeanSignal	SequencePulseAcquire
CardiacSegment	FracCorrect	MeanStd	SequenceSpaceEcho
CardiacSegmentGUI	FullGrad	MeanType	SequenceSpinEcho
CentralAxes	GenerateAmps	MedCouple	SequenceSteam
ChangeDwellTimeFid	GenerateBidsFileName	MedFilter	SequenceTSE
ChangeNetDimensions	GenerateBidsFolderName	MedianNoZero	ShiftedFourier
CheckDataDiscription	GenerateBidsName	MemoryUsage	ShiftedInverseFourier
CheckFile	GenerateGradients	MergeJSON	ShiftPulseProfile
CheckSegmentation	GenerateGradientsGUI	MergeSegmentations	ShiftSpectra
ClassDecoder	GenerateRotationFrames	MonitorCalc	ShowLog

4 All-Functions.nb AcquisitionMethod	DivonMackTharabhald	May Cood Doints	DoiostMan	_
	DixonMaskThreshhold	MaxSeedPoints	RejectMap	
ActivationBackground	DixonNucleus	MaxTracts	ReportFits	
ActivationIterations	DixonPhases	MaxTrainingRounds	RescaleMethod	
ActivationOutput	DixonPrecessions	MeanMethod	RescaleRecon	
ActivationSize	DixonRelaxivity	MeanRes	Resolutions	
ActivationThreshold	DixonTollerance	Method	ResolutionsA	
ActivationType	DownsampleSchedule	MethodReg	ReverseData	
AffineDirections	DropoutRate	MethodRegA	ReverseDirection	
AllowSelfDependencies	DropSamples	Monitor	ReversePoints	
AnisoFilterSteps	DropSlices	MonitorCalc	ReverseSets	
AnisoItterations	EchoShiftData	MonitorInterval	RobustFit	
AnisoKappa	EPGCalibrate	MonitorIVIMCalc	RobustFitParameters	
AnisoKernel	EPGFatShift	MonitorTagging	RotateGradients	
AnisoStepTime	EPGFitFat	MonitorUnwrap	RotationCorrect	
AnisoWeightType	EPGFitPoints	MotionCorrectSets	RoundLength	
ApodizationFunction	EPGMethod	NetworkArchitecture	RowSize	
AspectRatio	EPGMethodCal	NetworkDepth	Runs	
AugmentData	EPGRelaxPars	NiiDataType	Scaling	
AxesLabel	EPGSmoothB1	NiiLegacy	SegmentationMethod	
AxesMethod	FatFieldStrength	NiiMethod	SelectSubjects	
B1EqualPower	FeatureSchedule	NiiOffset	SenseRescale	
B1FilterData	FiberAngle	NiiScaling	SettingSchedule	
B1Masking	FiberLengthRange	NiiSliceCode	ShowMetric	
B1MaxPower	FieldStrength	NNThreshhold	ShowOutliers	
B1Output	FileType	NoiseSize	ShowPlot	
B1Scaling	Filling	NoiseType	SimNucleus	
B1ShimMethod	FilterMaps	NormalizeDensity	SliceRange	
BackgroundValue	FilterShape	NormalizeIVIM	SliceRangeSamples	
BasisSequence	FilterSize	NormalizeMethod	SmartMaskOutput	
BatchSize	FilterType	NormalizeOutputSpectra	SmartMethod	
BidsIncludeSession	FindTransform	NormalizeOverlap	SmoothHelix	
BidsTractographyMethod	FineTuneFit	NormalizeSets	SmoothItterations	
BlockType	FitConstrains	NormalizeSignal	SmoothSNR	
BloodMaskRange	FitFunction	NumberSamples	SortVecs	
Boxed	FitLineShape	NumberSamplesA	SparID	
BsplineDirections	FitOrder	OrderSpan	SparName	
BsplineSpacing	FitOutput	OutlierIncludeZero	SparOrientation	
BullPlotMethod	FittingOrder	OutlierIterations	SpectraBandwith	
CenterFrequency	FitTractSegments	OutlierMethod	SpectraFieldStrength	
CenterRange	FixPseudoDiff	OutlierOutput	SpectraNucleus	
CenterVoxel	FixPseudoDiffSD	OutlierRange	SpectraOutputPlots	
ChainSteps	FlipAxes	OutputCalibration	SpectraPpmShift	
Channels	FlipBvec	OutputCheckImage	SpectraSamples	
Classes	FlipGrad	OutputCoilSurface	SpectraSpacing	
CleanUpSegmentations	FullOutput	OutputForm	SphereColor	
ClippingStyle	FullSphere	OutputImage	SphereSize	
CoilArrayPlot	GetMaskOnly	OutputLabels	SplineDegree	
CoilSamples	GetMaskOutput	OutputPlot	SplineKnotsNumber	
CoilSurfaceVoxelSize	GOutput	OutputSamples	SplineRegularization	
ColorFunction	GradType	OutputSense	SplineSpacingFactor	
ColorValue	GRegularization	OutputSNR	SplitMethod	
CompressNii	GridLines	OutputSNR OutputTransformation	StartPoints	
ConditionColle	Gridines	Output Tails for macton	Star troints	

CardiacTools

Functions

BullseyePlot
CalculateWallMap
CardiacCoordinateSystem
CardiacSegment
CardiacSegmentGUI
CentralAxes
CreateHeart

ECVCalc
ExcludeSlices
GetMaskSegmentPoints
GetSegmentLines
GetSegmentSlices
HelixAngleCalc
LinesToSegmentIndex

MakeECVBloodMask
MakeLineImage
MakeMaskImage
MaskHelix
MaskToLines
PlotSegmentMask
PlotSegments

RadialSample SegmentLinesToMask SegmentsPerSlice TransmuralPlot

Options

AxesMethod
BackgroundValue
BloodMaskRange
BullPlotMethod
ColorFunction
CutOffMethod
DistanceMeasure
DropSamples

GridLineSpacing
GroupPerSegment
ImageSize
LCMMethod
MakeSegmentPlots
MaskWallMap
Method
OutputCheckImage

PlotLabel
PlotRange
PlotStyle
RadialSamples
ReverseDirection
ReversePoints
RowSize
SegmentationMethod

ShowOutliers
ShowPlot
SmoothHelix
StartPoints
StartSlices
TextNumberForm
TextOffset
TextSize

CoilTools

Functions

CoilSNRCalc FindCoilPosition LoadCoilSetup LoadCoilTarget MakeCoilLayout MakeNoisePlots MakeWeightMask

Options

CoilArrayPlot CoilSurfaceVoxelSize ColorFunction ImageSize OutputCoilSurface PlotRange

DenoiseTools

Functions

AnisoFilterTensor DenoiseCSIdata DenoiseDynamicSpectraData

NNDeNoise

Options

AnisoFilterSteps AnisoWeightType MonitorCalc PCANoiseSigma AnisoItterations DeNoiseIterations NNThreshhold PCAOutput AnisoKappa DeNoiseKernel PCATollerance **PCAClipping** PCAComplex AnisoKernel DeNoiseMonitor **PCAWeighting PCAKernel** AnisoStepTime Method

DixonTools

Functions

DixonPhaseFixDixonFlipsUnwrapDixonReconstructGenerateAmpsUnwrapDCTDixonToPercentOptimizeDixonEchoUnwrapListFindInPhaseEchosSimulateDixonSignalUnwrapSplit

Options

DixonAmplitudes DixonFilterOutput DixonMaskThreshhold DixonBipolar DixonFilterSize DixonNucleus DixonClipFraction DixonFilterType DixonPhases DixonConstrainPhase DixonFitPhase DixonPrecessions DixonCorrectT1 DixonFixT2 DixonRelaxivity DixonFieldStrength DixonFrequencies DixonTollerance DixonFilterInput DixonIterations MaxIterations

Wrap

PCADeNoise

WeightMapCalc

MonitorCalc MonitorUnwrap PhaseEchos UnwrapDimension UnwrapThresh

ElastixTools

Functions

ReadTransformParameters RegisterCardiacData RegisterData RegisterDataSplit RegisterDataTransform RegisterDataTransformSplit RegisterDiffusionData RegisterDiffusionDataSplit RegisterTensorData TransformData \$debugElastix

Options

AffineDirections
BsplineDirections
BsplineSpacing
DeleteTempDirectory
FindTransform
HistogramBins
HistogramBinsA

InterpolationOrderReg InterpolationOrderRegA Iterations IterationsA Method MethodReg MethodRegA

NumberSamples
NumberSamplesA
OutputImage
OutputTransformation
PCAComponents
PrintTempDirectory
RegistrationTarget

Resolutions ResolutionsA ShowMetric SplitMethod TempDirectory TransformMethod

UseGPU

FasciculationTools

Functions

AnalyzeActivations

EvaluateActivation

FindActivations

SelectActivations

Options

ActivationBackground ActivationIterations ActivationOutput ActivationSize ActivationThreshold IgnoreSlices

MaskDilation ThresholdMethod

GeneralTools

Functions

ApplyCrop FileSelect MedFilter RotateDimensionsLeft AutoCropData FindCrop MedianNoZero RotateDimensionsRight BSplineCurveFit FindMaxDimensions MemoryUsage RotationMatrixToQuaternion ClearTemporaryVariables GetAssetLocation MonitorCalc RotationMatrixToOuaternionVector

CompilableFunctionsGridDataNiiFileExistQSaveImageConvertExtensionGridData3DNNLeastSquaresSignNoZeroCropDataGyromagneticRatioPadToDimensionsSqueeze

CutDataLapFilterQMRIToolsFuncPrintStandardDeviationNoZeroDataToVectorLLeastSquaresQMRIToolsFunctionsStdFilter

DecomposeScaleMatrix LogNoZero QMRIToolsPackages StichData
DecomposeScaleMatrix MADNoZero QuaternionToRotationMatrix StringPadInteger

DevideNoZeroMakeCoordinatesQuaternionVectorToRotationMatrixSumOfSquaresDynamicPartitionMakeFunctionGraphRescaleDataTensMatEmptyDirectoryQMakeIntFunctionReverseCropTensVecExpNoZeroMBCountReverseDimensionsVectorToData

ExtractDemoData MeanNoZero RMSNoZero

Options

AllowSelfDependencies InterpolationOrder PadValue CropOutput LabelPlacement CenterRange CropPadding SplineDegree CenterVoxel OutputWeights SplineKnotsNumber FileType CropAlways ImageResolution Padding SplineRegularization

CropInit ImageSize PadDirection WindowTitle

GradientTools

Functions

Bmatrix ConditionNumberCalc **FullGrad** GradSeq BmatrixCalc ConvertGrads GenerateGradients ImportGradObj BmatrixConv CorrectBmatrix GenerateGradientsGUI OverPlusCalc BmatrixInv GetGradientScanOrder SelectBvalueData CorrectGradients BmatrixRot GetSliceNormal UniqueBvalPosition EnergyCalc BmatrixToggle FinalGrads GetSliceNormalDir

CalculateMoments FindOrder GradBmatrix

Options

ConditionCalcMethodPhaseEncodingUnitMultiFlipAxesMethodRegRunsUseGradFlipGradOrderSpanStepsVisualOpt

FullSphere OutputPlot StepSizeI GradType OutputType SwitchAxes

IVIMTools

Functions

BayesianIVIMFit2FConvertiIVIMCorrectDataThetaConvBayesianIVIMFit3FracCorrectIVIMFunctionThetaConviCorrectParMapHistogramParIVIMResidualsFConvertIVIMCalcMeanBvalueSignal

Options

ChainStepsFitConstrainsIVIMConstrainsOutputSamplesCorrectParFixPseudoDiffIVIMFixedParallelizeFilterMapsFixPseudoDiffSDIVIMTensFitUpdateStepFilterSizeIVIMComponentsMethod

FilterType IVIMConstrained MonitorIVIMCalc

JcouplingTools

Functions

GetSpinSystemSequenceSpinEchoSimEvolveSimSignalMakeSpinSystemSequenceSteamSimHamiltonianSimSpoilSequencePulseAcquireSequenceTSESimReadoutSysTableSequenceSpaceEchoSimAddPhaseSimRotate

Options

CenterFrequency LinewidthShape ReadoutOutput SimNucleus FieldStrength ReadoutBandwith ReadoutPhase Linewidth ReadoutMethod ReadoutSamples

LoggingTools

Functions

AddToLog ExportDirectoryTree MakeCheckFile ShowLog CheckFile ExportLog PrintDirectoryTree DirectoryTree ImportLog ResetLog

Options

MaskingTools

Functions

DilateMask
FitGradientMap
GetCommonSegmentation
GetSegmentationLabels
HomogenizeData
Mask

MaskData
MaskSegmentation
MergeSegmentations
NormalizeData
NormalizeMeanData
RemoveMaskOverlaps

ReplaceSegmentations
RescaleSegmentation
SegmentationVolume
SegmentMask
SelectMaskComponents
SelectSegmentations

SmoothMask SmoothSegmentation SplitSegmentations

Options

FitOrder MaskClosing MaskComponents MaskDilation MaskFiltKernel MaskSmoothing NormalizeMethod SmoothItterations

MuscleBidsTools

Functions

AddToJson
BidsDcmToNii
CheckDataDiscription
ExtractFromJSON
GenerateBidsFileName
GenerateBidsFolderName

GenerateBidsName GetConfig GetJSONPosition ImportJSON MergeJSON MuscleBidsConvert

MuscleBidsMerge MuscleBidsProcess MuscleBidsSegment MuscleBidsTractography PartitionBidsFolderName PartitionBidsName SelectBidsFolders SelectBidsSessions SelectBidsSubjects ViewConfig

Options

BidsIncludeSession BidsTractographyMethod DeleteAfterConversion SelectSubjects VersionCheck

NiftiTools

Functions

CompressNiiFiles
CorrectNiiOrientation
DcmToNii
ExportBmat
ExportBval
ExportBvec

ExportNii
ExtractNiiFiles
GetNiiOrientation
ImportBmat
ImportBval
ImportBvalvec

ImportBvec
ImportExploreDTItens
ImportNii
ImportNiiDiff
ImportNiiDix
ImportNiiT1

ImportNiiT2
MakeNiiOrentationQ
MakeNiiOrentationS

Options

CompressNii DeleteOutputFolder FlipBvec Method MonitorCalc NiiDataType NiiLegacy NiiMethod NiiOffset NiiScaling NiiSliceCode PositiveZ

RotateGradients UseSubfolders UseVersion

Functions

ColorFAPlot Link3DGraphic PlotData PlotMoments GenerateRotationFrames PlotData3D PlotSegmentations ListSpherePlot PlotDefGrid GetSliceData MakeSliceImages PlotSequence GetSlicePositions PlotContour PlotDuty GradientPlot PlotCorrection PlotIVIM

Options

ClippingStyle ContourScaling MakeCheckPlot PlotSpace ColorFunction ContourSmoothRadius Method PositiveZ ContourColor DropSlices NormalizeIVIM RandomizeColor ContourColorRange ImageLegend PeakNumber SphereColor ContourOpacity ImageOrientation PlotColor SphereSize ContourResolution ImageSize PlotRange

ProcessingTools

Functions

B1MapCalc FindOutliers JoinSets RotateData B1Shimming FitData MeanRange RotateTensor CombineB1 GetMaskData MeanSignal SmartMask CorrectJoinSetMotion GetMaskMeans MeanStd SNRCalc DataTransformation GetTractMeans MedCouple SNRMapCalc DatTot Hist NumberTableForm SplitSets DatTotXLS Hist2 ParameterFit

InvertDataset

Options

ErrorPlot

AxesLabel GetMaskOutput OutlierIterations SmartMaskOutput B1EqualPower ImageSize OutlierMethod SmartMethod B1FilterData InterpolationOrder OutlierOutput SmoothSNR B1Masking JoinSetSplit OutlierRange Strictness B1MaxPower MaskCompartment OutputSNR TableAlignments B10utput MeanMethod PaddOverlap TableDepth B1Scaling Method PadOutputDimensions TableDirections B1ShimMethod MonitorCalc PlotLabel TableHeadings ColorValue MotionCorrectSets ReferenceB1 TableMethod FitFunction NormalizeOverlap ReverseData TableSpacing FitOutput NormalizeSets ReverseSets UseMask GetMaskOnly OutlierIncludeZero Scaling

ParameterFit2

ReconstructionTools

Functions

CoilCombine FourierRescaleData MakeHammingFilter CoilWeightedRecon FourierShift MakeSense CoilWeightedReconCSI FourierShifted MeanType DeconvolveCSIdata HammingFilterCSI NoiseCorrelation FourierKspace2D HammingFilterData NoiseCovariance FourierKspace3D InverseFourierShift NormalizeSpectra FourierKspaceCSI InverseFourierShifted OrderKspace

ReadListData
SagitalTranspose
ShiftedFourier
ShiftedInverseFourier

TotalType

Options

AcquisitionMethod EchoShiftData CoilSamples HammingFilter

DeconvolutionMethod Method

NormalizeOutputSpectra

OutputSense ReconFilter RescaleRecon SenseRescale

WienerRegularization

RelaxometryTools

Functions

CalibrateEPGT2FitEPGSignalShiftPulseProfileT2FitCreateT2DictionaryEPGT2FitT1FitTriExponentialT2FitDictionaryMinSearchNonLinearEPGFitT1rhoFit

Options

DictB1RangeEPGCalibrateEPGMethodCalDictT2fRangeEPGFatShiftEPGRelaxParsDictT2fValueEPGFitFatEPGSmoothB1DictT2IncludeWaterEPGFitPointsMethodDictT2RangeEPGMethodMonitorCalc

OutputCalibration WaterFatShift

WaterFatShiftDirection

ScientificColorData

Functions

AddScientificColors ExtractColorData

Options

SegmentationTools

Functions

AddLossLayer AnalyseNetworkFeatures ApplySegmentationNetwork AugmentImageData AugmentTrainingData ChangeNetDimensions CheckSegmentation ClassDecoder ClassEncoder ClassifyData DataToPatches

DiceLossLayer DiceSimilarity FocalLossLayer GetNeuralNet GetTrainData ImportITKLabels JaccardLossLayer JaccardSimilarity MakeChannelClassGrid MakeChannelClassImage MakeChannelImage

MakeClassifyImage MakeClassifyNetwork MakeClassImage MakeDistanceMap MakeNode MakeUnet MuscleLabelToName MuscleNameToLabel NetDimensions NetSummary PatchesToData

PrepareTrainingData SegmentData SegmentDataGUI ShowTrainLog SplitDataForSegementation SurfaceDistance TrainSegmentationNetwork TverskyLossLayer

Options

Channels Classes Dimensions ActivationType AugmentData BatchSize BlockType CleanUpSegmentations

DataPadding DataTag

DistanceRange DownsampleSchedule DropoutRate FeatureSchedule ImageSize InputLabels L2Regularization LabelTag

LearningRate LoadTrainingData

LossEunction MaxPatchSize MaxTrainingRounds Method Monitor MonitorCalc MonitorInterval NetworkArchitecture NetworkDepth OutputLabels

PatchesPerSet PatchNumber PatchPadding PatchSize RescaleMethod RoundLength SettingSchedule TargetDevice TestRun

\$debugUnet

SimulationTools

Functions

AddNoise BlochSeries CalculateGfactor CreateDiffData ErnstAngle

GESignal GetPulseProfile GfactorSimulation PlotSimulation PlotSimulationAngle

PlotSimulationAngleHist PlotSimulationHist PlotSimulationVec Pulses Signal

SimAngleParameters SimParameters SimulateDualTR SimulateSliceEPG Tensor

Options

FatFieldStrength GOutput GRegularization MagnetizationVector NoiseSize NoiseType PlotRange Reject

ReportFits SliceRange SliceRangeSamples SortVecs

TensOutput

SpectroTools

Functions

ApodizeEcho
ApodizeFid
ApodizePadEcho
ApodizePadFid
ApodizePadSpectra
ApodizeSpectra
ChangeDwellTimeFid
CompareFidFitPlot
CompareSpectraFitPlot
CorrectTEFid

CorrectTESpec
CSIInterface
ExportSparSdat
FindSpectraPpmShift
FitSpectra
FitSpectraResultTable
GetGyro
GetPpmRange

GetSpectraBasisFunctions GetTimePpmRange GetTimeRange PlotSpectra
ImportSparSdat ReadjMRUI
MakeSpectraResultPlot ShiftSpectra
PadEcho SpectraFitResult
PadFid TimeShiftEcho
PadSpectra TimeShiftFid
PhaseCorrectSpectra TimeShiftFidV
PhaseShiftSpectra

Options

ApodizationFunction
AspectRatio
BasisSequence
CenterFrequency
Filling
FineTuneFit
FitLineShape
GridLines

GridLineSpacing
ImageSize
InitializeFit
Method
PaddingFactor
PlotColor
PlotLabel
PlotLabels

PlotRange
ReadoutType
SparID
SparName
SparOrientation
SpectraBandwith
SpectraFieldStrength
SpectraNucleus

PlotCSIData

PlotFid

SpectraOutputPlots SpectraPpmShift SpectraSamples SpectraSpacing SplineSpacingFactor

TaggingTools

Functions

AnnalyzeTagging

CalculateDispacementParameters

Options

HistoryWeighting

MonitorTagging

TensorTools

Functions

ADCCalc ECalc
AngleCalc EigensysCalc
AngleMap EigenvalCalc
ConcatenateDiffusionData EigenvecCalc
Correct ExpTensor
Deriv FACalc

DriftCorrect FlipGradientOrientation SortDiffusionData

TensorCalc TensorCorrect TransformTensor WestinMeasures

Options

Distribution Method PerformanceGoal RobustFitParameters
FilterShape MonitorCalc Reject RotationCorrect
FullOutput NormalizeSignal RejectMap UseMask
MeanRes Parallelize RobustFit

FlipTensorOrientation

LogTensor

ParameterCalc

ResidualCalc

SigmaCalc

RemoveIsoImages

TractographyTools

Functions

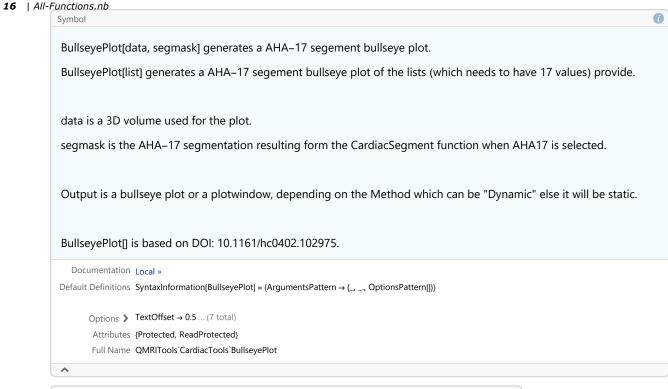
ExportTracts FitTracts PlotTracts TractAngleMap FiberLength GetTractValues ResampleTracts TractDensityMap FiberTractography ImportTracts RescaleTracts TractLengthMap FilterTracts MoveTracts SeedDensityMap FindTensorPermutation PlotSegmentedTracts SegmentTracts

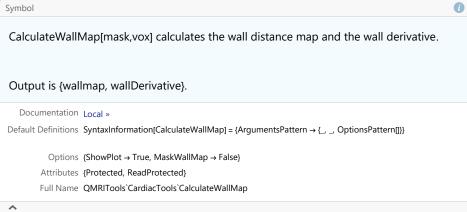
Options

Boxed ImageSize NormalizeDensity TensorPermutations ColorFunction InterpolationOrder OutputForm TracMonitor MaxSeedPoints PerformanceGoal FiberAngle TractColoring FiberLengthRange MaxTracts StepSize TractReduction FittingOrder Method StopThreshhold TractScaling FitTractSegments TensorFilps TractSize Monitor

In[@]:= QMRIToolsFuncPrint[]

CardiacTools





```
CardiacCoordinateSystem[mask, vox] creates the cardiac coordinate system within the mask and is used in HelixAngleCalc.

Output is a set of vectors {radvecn, norvecc, cirvec}, being the radial, normal and circular axes of each voxel respectivley.

If the option showPlot is true the output is {{radvecn, norvecc, cirvec}, plots}.

Documentation Local »

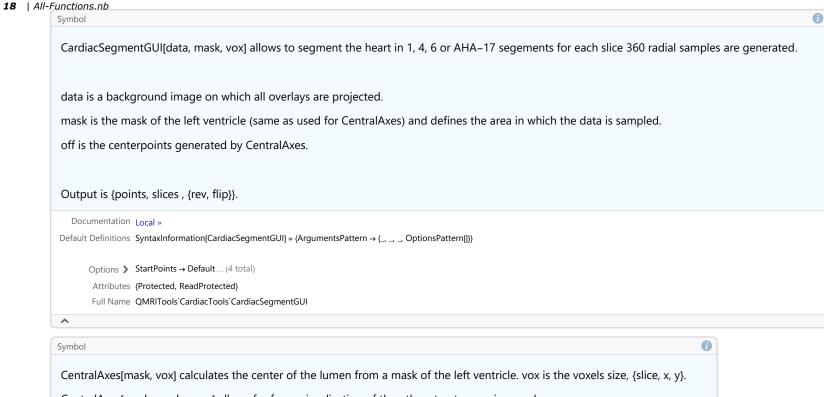
Default Definitions SyntaxInformation[CardiacCoordinateSystem] = {ArgumentsPattern → {_ , _ , OptionsPattern[]}}

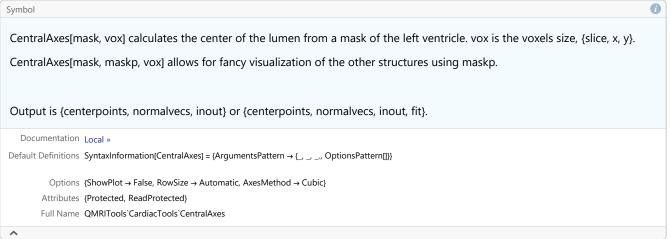
Options {ShowPlot → False, LCMMethod → WallMap, AxesMethod → Quadratic}

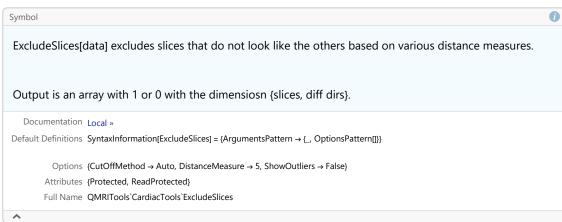
Attributes {Protected, ReadProtected}

Full Name QMRITools'CardiacTools'CardiacCoordinateSystem
```

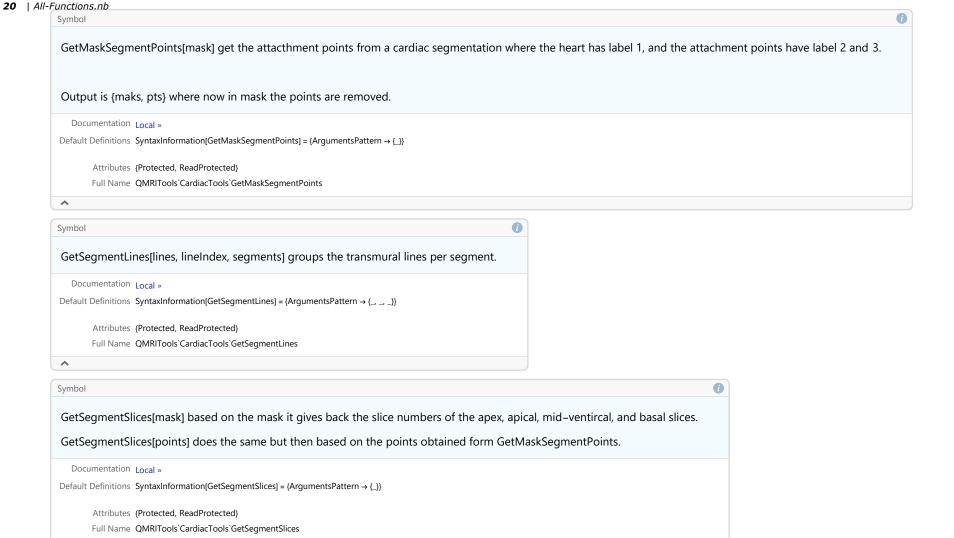
^







All-Functions.nb | 19



0

HelixAngleCalc[eigenvectors, mask, vox] calculates the helix angle matrix of cardiac data using only a left ventricle mask.

HelixAngleCalc[eigenvectors, mask, maskp, vox] calculates the helix angle matrix of cardiac data using only a left ventricle mask, and a maskp for visualization.

HelixAngleCalc[eigenvectors, mask, centerpoint, vec, inout, vox] calculates the helix angle matrix of cardiac data using only a left ventricle mask.

HelixAngleCalc[eigenvectors, mask, maskp, centerpoint, vec, inout, vox] calculates the helix angle matrix of cardiac data using a left vantricle mask and a maskp for visualization.

eigenvectors are the tensor eigenvectors calculated with EigenvecCalc.

mask is a mask of the left ventricle.

maskp is a mask used for visualization.

vox is the voxels size, {slice, x, y}.

The following values are calculated automaticlay Using CentralAxes but can also be provided as an input.

centerpoint is the center of each slice calculated with CentralAxes.

inout is the inner and outer radius calcualted with CentralAxes.

vec is the vector describin the central axes of the heart, calculated with CentralAxes.

Output is het fiber angle matrix $FAM = \{9, slice, x, y\}$ or $\{FAM, plot\}$.

The angles are in degrees.

HelixAngleCalc[] is based on DOI: 10.1186/1532-429X-17-S1-P15.

Documentation Local »

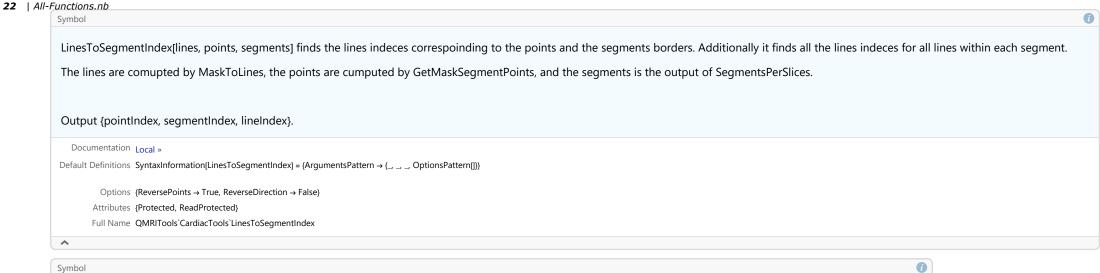
Default Definitions SyntaxInformation[HelixAngleCalc] = {ArgumentsPattern → {_, _, _, _, _, OptionsPattern[]}}

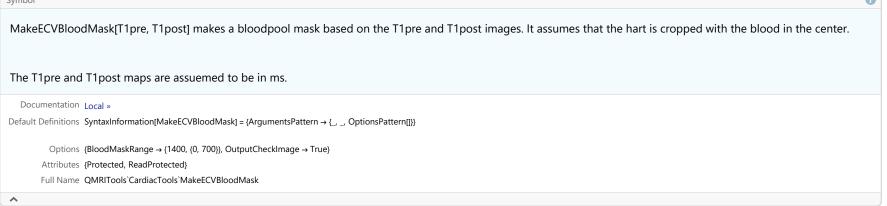
Options {ShowPlot → True, LCMMethod → WallMap, AxesMethod → Quadratic}

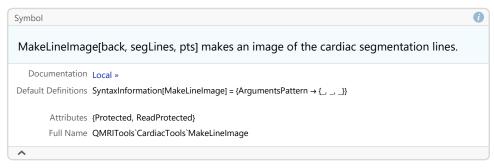
Attributes {Protected, ReadProtected}

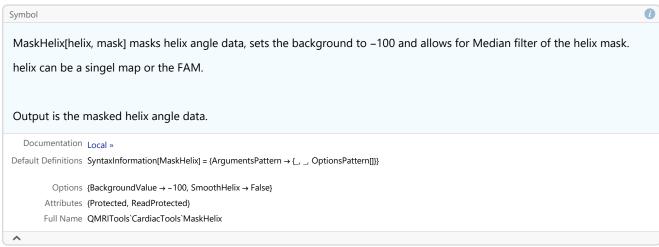
Full Name QMRITools`CardiacTools`HelixAngleCalc

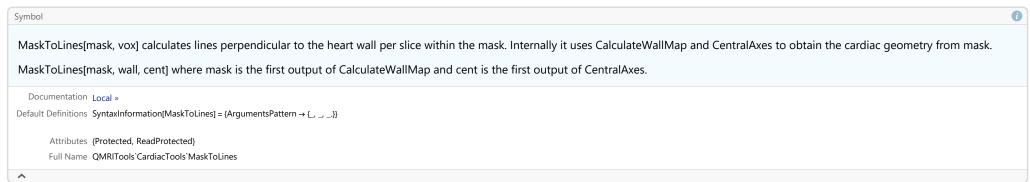
^

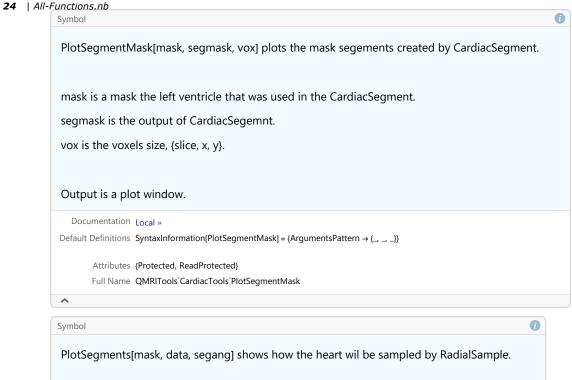


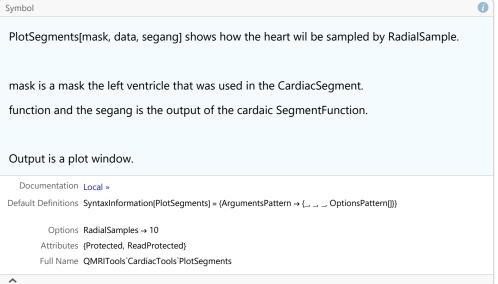




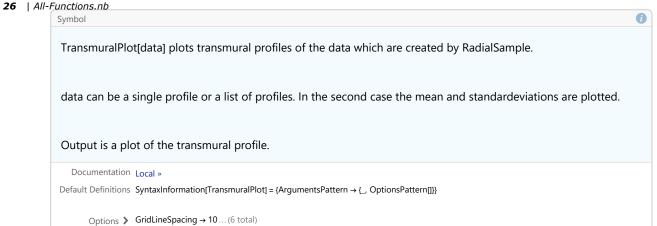








```
0
Symbol
RadialSample[mask, data, segang] radialy samples the provided parametermap data.
The mask should be a mask of the left ventricle that was used in the CardiacSegment.
segang is the output of the cardaic SegmentFunction.
Output is {points, vals} which are orderd as indicated by the user.
  Documentation Local »
Default Definitions SyntaxInformation[RadialSample] = {ArgumentsPattern → {_, _, _, _, OptionsPattern[]}}
         Options {RadialSamples → 10, DropSamples → 0}
       Attributes {Protected, ReadProtected}
       Full Name QMRITools`CardiacTools`RadialSample
\wedge
Symbol
SegmentLinesToMask[mask, segLines] cuts the mask based one the transural lines per segments which can be obtained by GetGesmentLines.
  Documentation Local »
Default Definitions SyntaxInformation[SegmentLinesToMask] = {ArgumentsPattern \rightarrow \{\_, \_\}}
       Attributes {Protected, ReadProtected}
       Full Name QMRITools`CardiacTools`SegmentLinesToMask
\wedge
Symbol
SegmentsPerSlice[points] gives the number of segments per slice where the slice distribution is determined by GetSegmentSlices.
SegmentsPerSlice[slices, points] does the same but the slices are given manually.
  Documentation Local »
Default Definitions SyntaxInformation[SegmentsPerSlice] = {ArgumentsPattern → {_, OptionsPattern[]}}
         Options {GroupPerSegment → True, SegmentationMethod → AHA}
       Attributes {Protected, ReadProtected}
       Full Name QMRITools`CardiacTools`SegmentsPerSlice
^
```

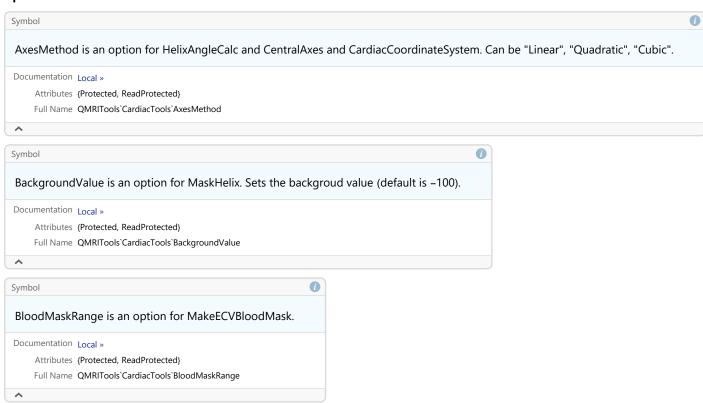


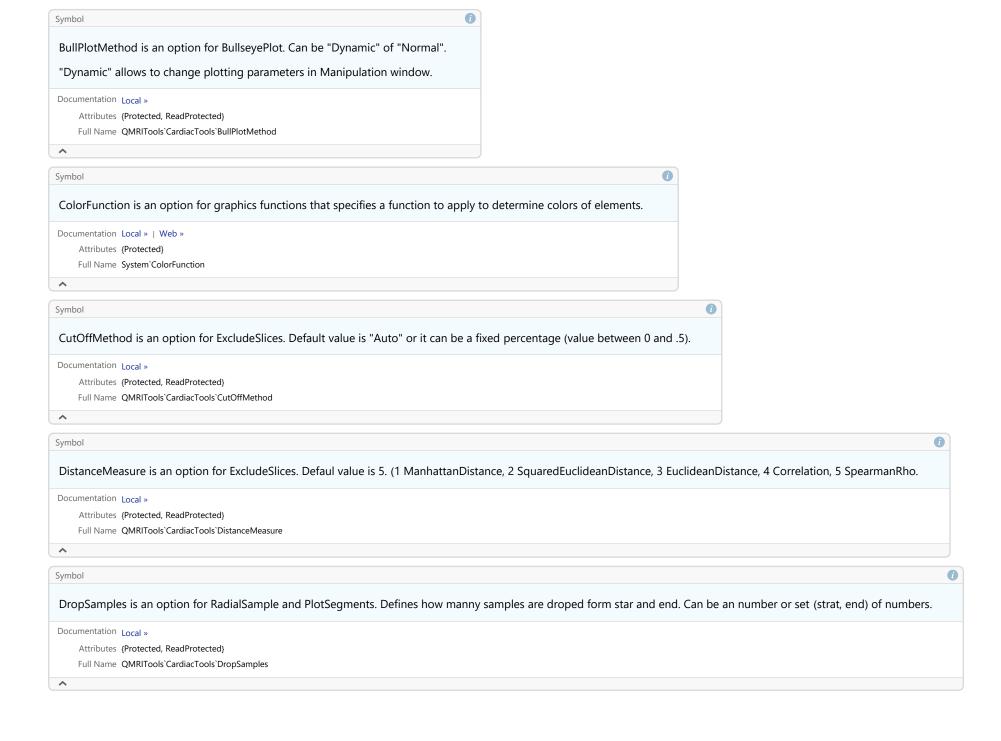
Options

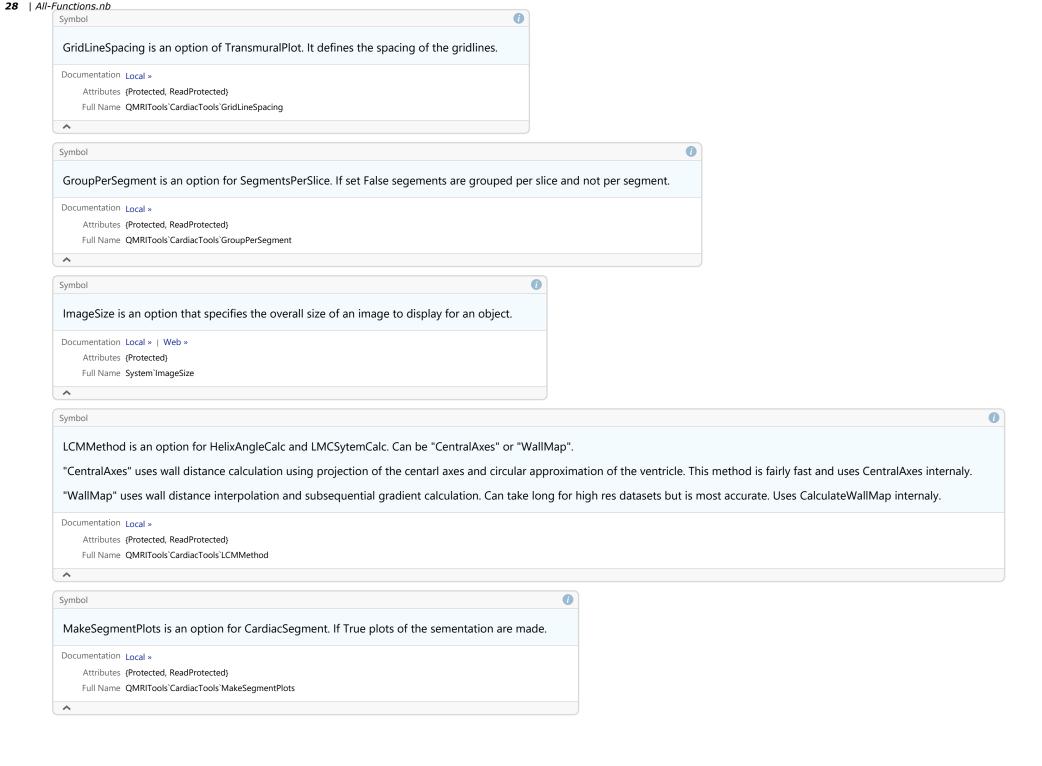
^

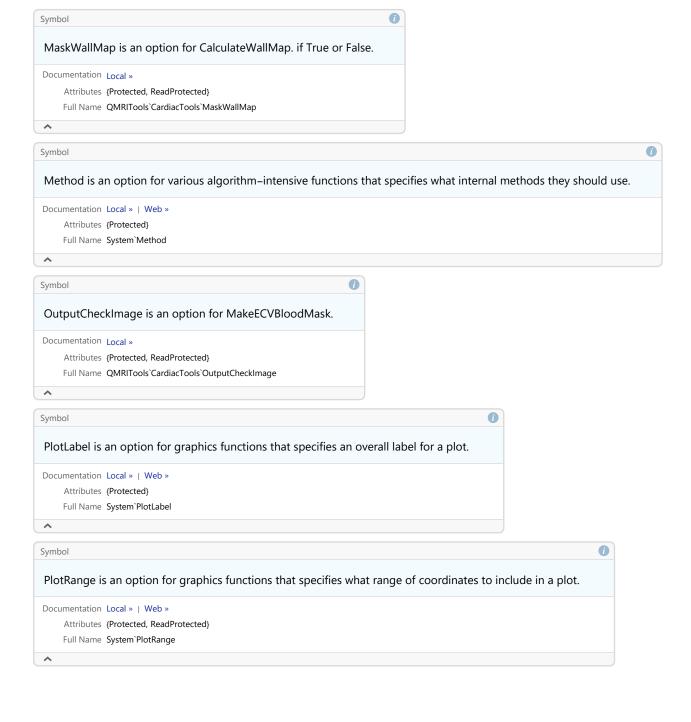
Attributes {Protected, ReadProtected}

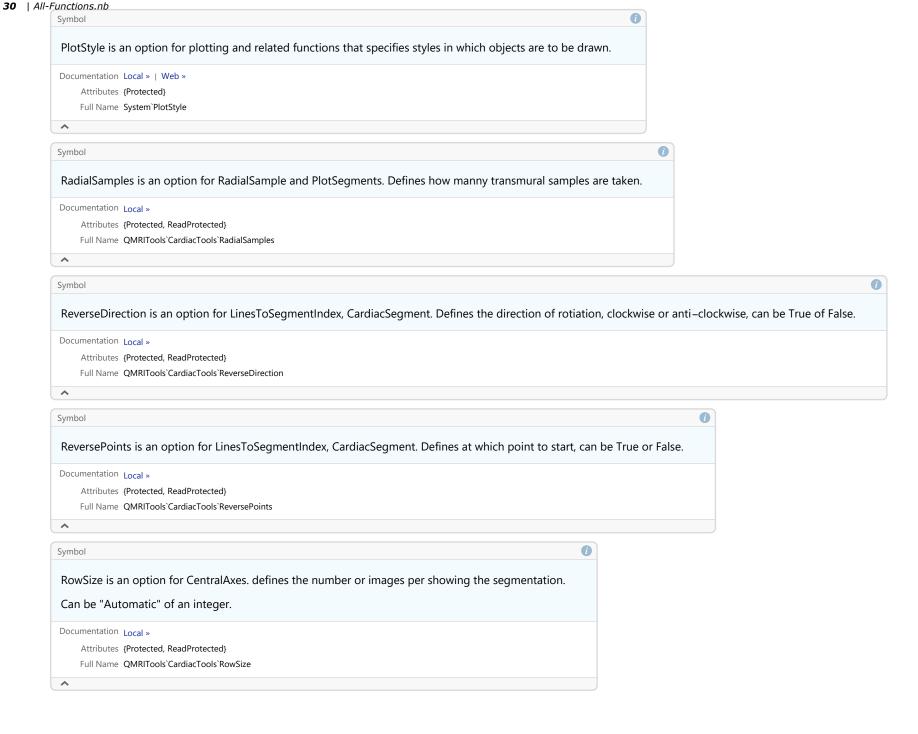
Full Name QMRITools`CardiacTools`TransmuralPlot

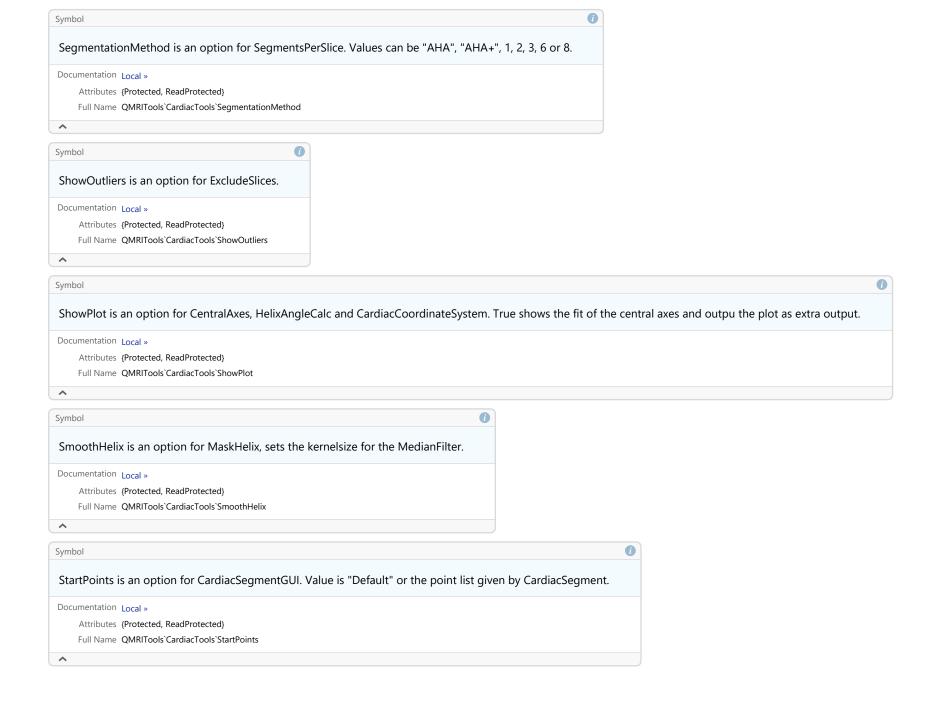


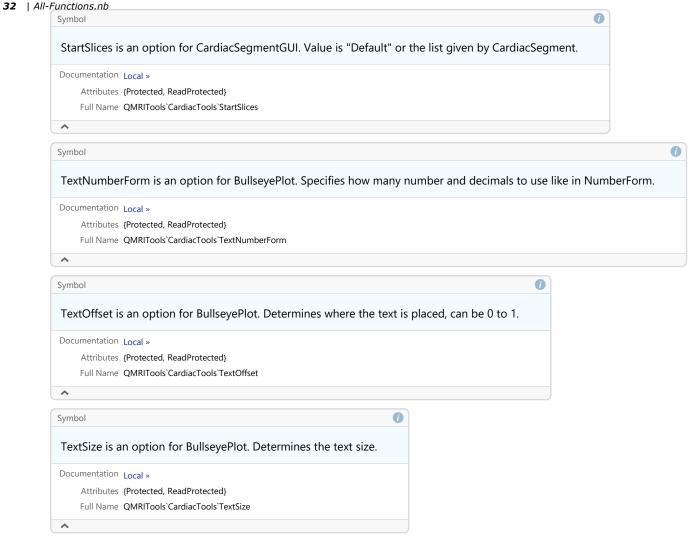






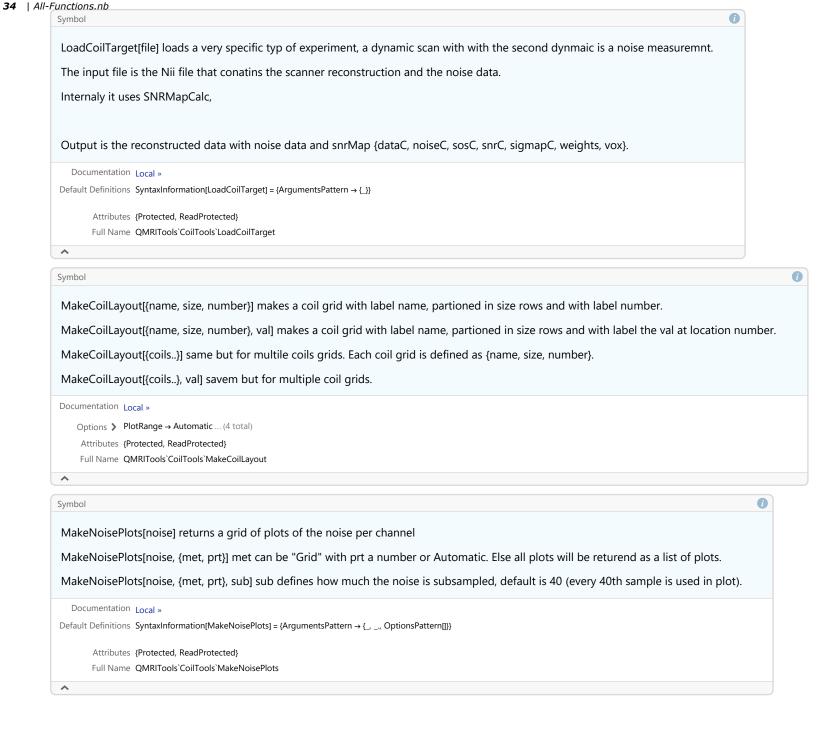




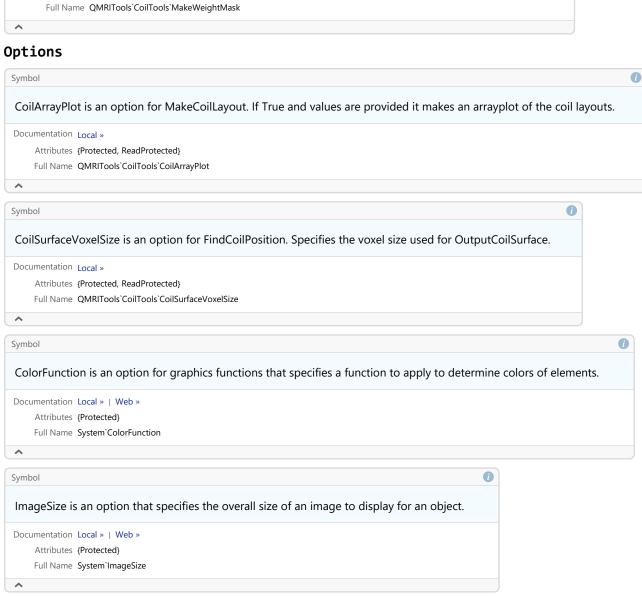


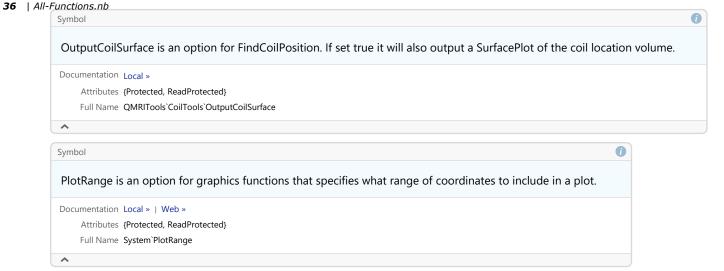
CoilTools

```
Symbol
CoilSNRCalc[coils, noise] calculates the sensitivity weighted snr of multiple coil elements using magnitude signal and noise.
Output is {data, noise, sos, snr, sigmap, weights}.
  Documentation Local »
Default Definitions SyntaxInformation[CoilSNRCalc] = {ArgumentsPattern \rightarrow \{\_, \_\}}
       Attributes {Protected, ReadProtected}
       Full Name QMRITools'CoilTools'CoilSNRCalc
^
Symbol
FindCoilPosition[weights] finds the coil posision by locating the highest intensity location in the coil weight map, which can be obtianed by LoadCoilSetup or SumOfSquares.
Internally it uses MakeWeightMask to remove the noise of the weightmasks.
FindCoilPosition[weights, mask] limits the search region to the provided mask.
  Documentation Local »
Default Definitions SyntaxInformation[FindCoilPosition] = {ArgumentsPattern → {_, _,, OptionsPattern[]}}
         Options {OutputCoilSurface \rightarrow False, CoilSurfaceVoxelSize \rightarrow {1, 1, 1}}
       Attributes {Protected, ReadProtected}
       Full Name QMRITools'CoilTools'FindCoilPosition
^
                                                                                                                                                                      0
Symbol
LoadCoilSetup[file] load a very specific type of coil experiment, a dynmic scan with a setup of which the second dynamic is a noise measurement.
The input file is the Nii file that conatins the individualy reconstruted coil images and the noise data.
Internaly it uses CoilSNRCalc and SumOfSquares.
Output is the coil data with coil noise data and snrmap based on the SumOfSquares addition, the SOS reconstruction and the SOS weights.
{dataC, noiseC, sosC, snrC, sigmapC, weights, vox}.
  Documentation Local »
Default Definitions SyntaxInformation[LoadCoilSetup] = {ArgumentsPattern \rightarrow {_, _.}}
       Attributes (Protected, ReadProtected)
       Full Name QMRITools`CoilTools`LoadCoilSetup
^
```

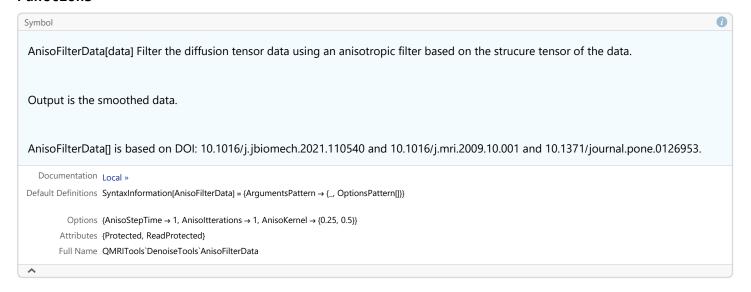


```
Symbol
MakeWeightMask[weights] creates a mask of homogeneous regions of weightmaps removing the noise.
  Documentation Local »
Default Definitions SyntaxInformation[MakeWeightMask] = \{ArgumentsPattern \rightarrow \{\_\}\}\
       Attributes {Protected, ReadProtected}
       Full Name QMRITools`CoilTools`MakeWeightMask
```





DenoiseTools



```
AnisoFilterTensor[tens, diffdata] Filter the tensor tens using an anisotropic diffusion filter (Perona–Malik).

It uses the diffusion weighted data diffdata to find edges that are not visible in the tensor.

Edge weights based on the diffusion data are averaged over all normalized diffusion direction.

AnisoFilterTensor[tens] Same but does not use the data for edge identification.

Output is the smoothed tensor.

AnisoFilterTensor[] is based on DOI: 10.1109/ISBI.2006.1624856.

Documentation Local »

Default Definitions SyntaxInformation[AnisoFilterTensor] = (ArgumentsPattern → { _ _ _ OptionsPattern[])}

Options > AnisoWeightType → 2 ... (4 total)

Attributes (Protected, ReadProtected)

Full Name QMRITools DenoiseTools AnisoFilterTensor
```



DenoiseDynamicSpectraData[spectra] perfroms PCA denoising of the complex values spectra, The data is given as a list of dynamicly acquired spectra {dynamic ,spectra}.

Documentation Local »

Default Definitions SyntaxInformation[DenoiseDynamicSpectraData] = {ArgumentsPattern → {_}}}

Attributes {Protected, ReadProtected}

Full Name QMRITools'DenoiseTools'DenoiseDynamicSpectraData

NNDeNoise[data] removes rician noise from the data using self supravized neural net.

NNDeNoise[data, mask] removes rician noise from the data with PCA using self supravized neural net withing the mask.

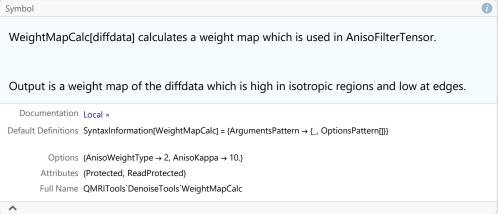
PCADeNoise[] is based on DOI:10.48550/arXiv.2011.01355.

Documentation Local >

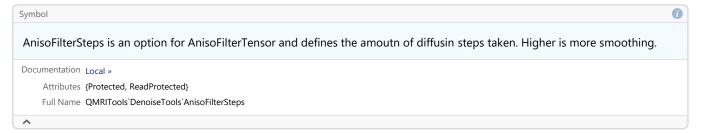
Default Definitions SyntaxInformation[NNDeNoise] = {ArgumentsPattern → {_, _, _, _, _, OptionsPattern[]}}

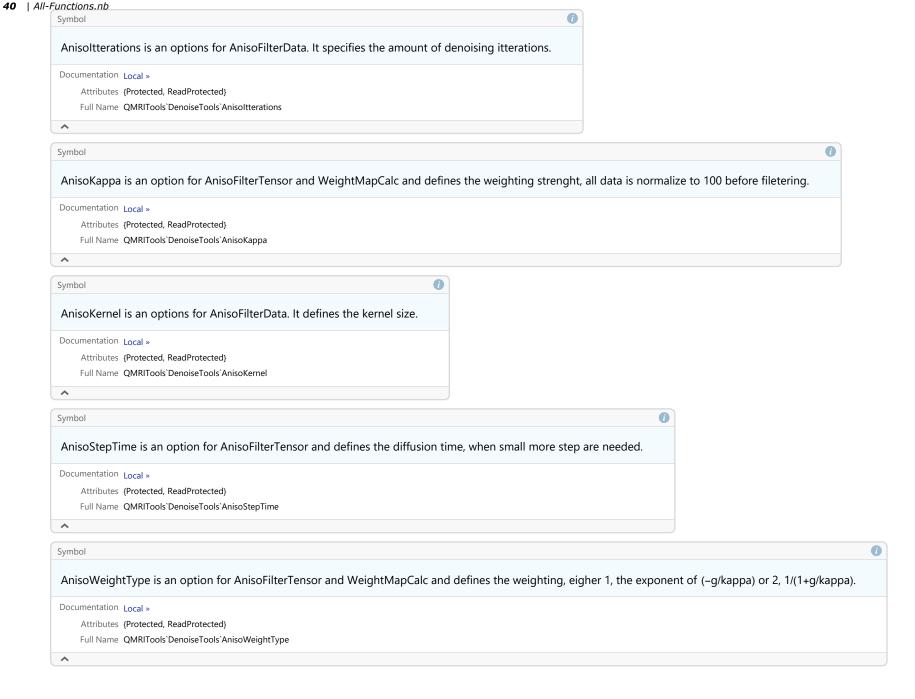
Options NNThreshhold → 2

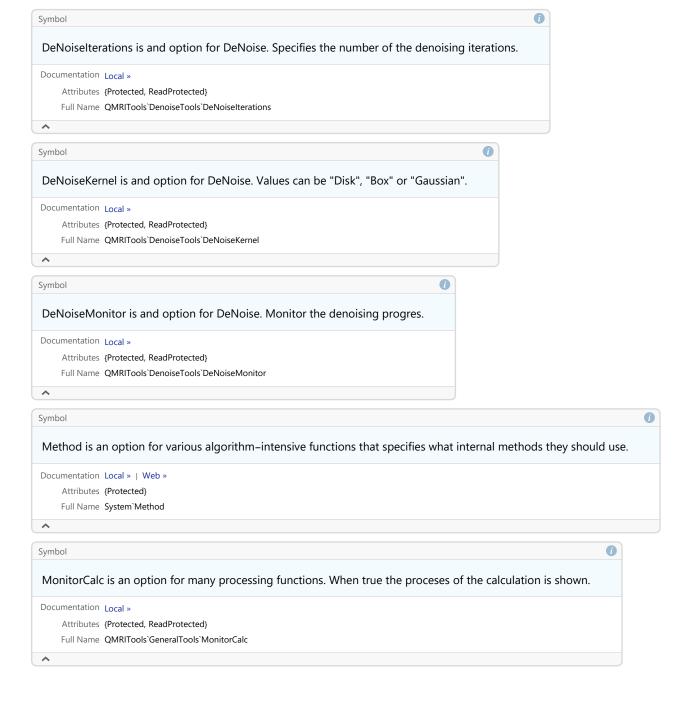
Attributes {Protected, ReadProtected}}
Full Name QMRITools'DenoiseTools'NNDeNoise

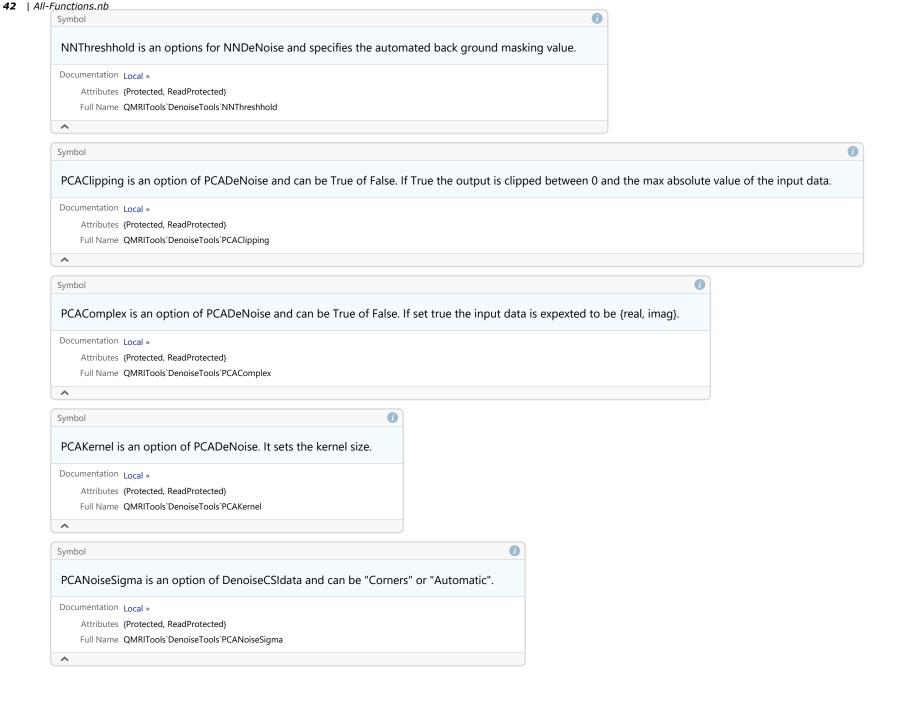


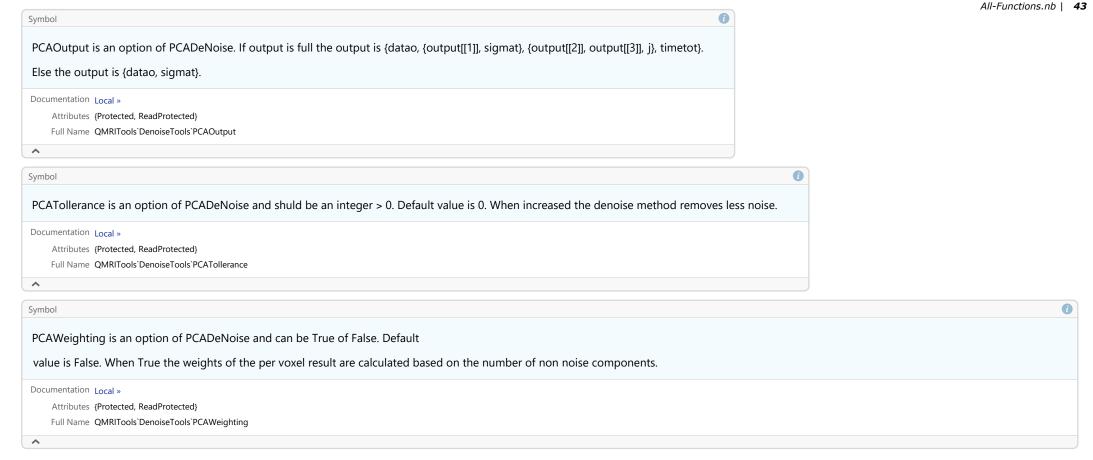
Options



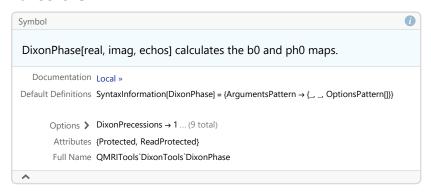


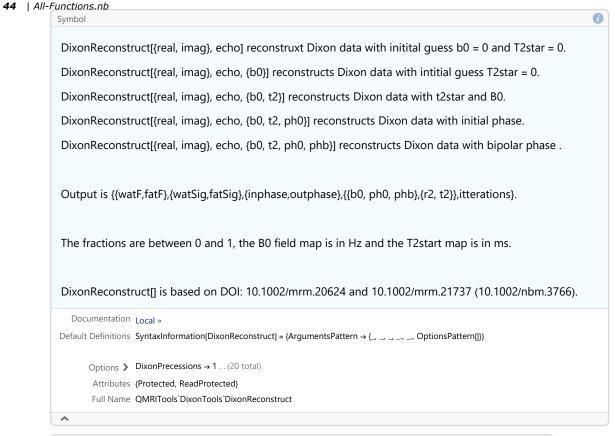


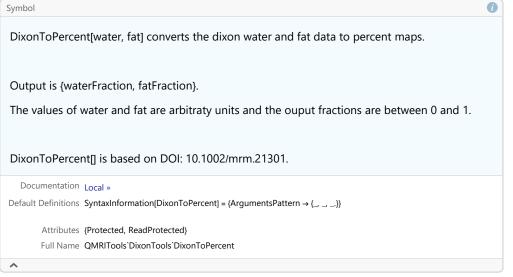




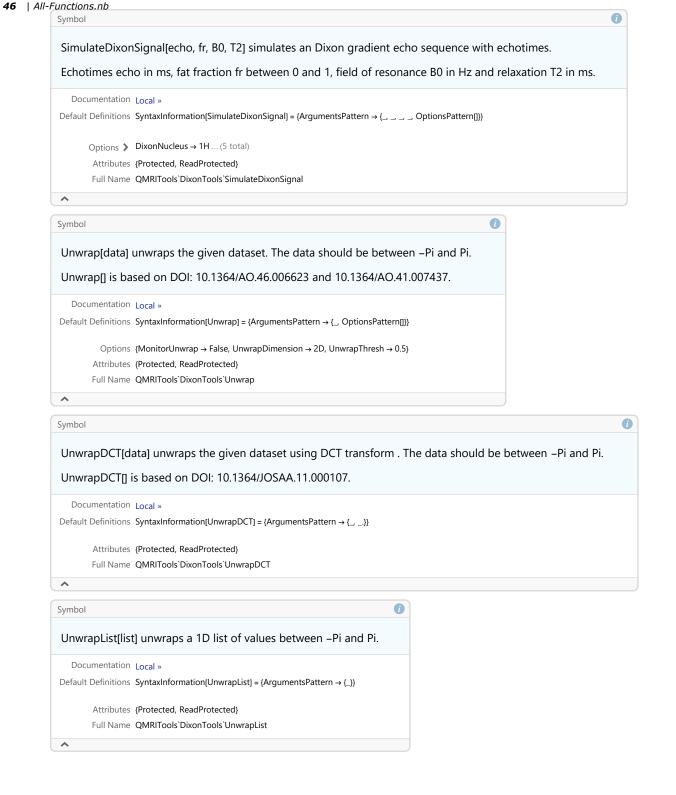
DixonTools

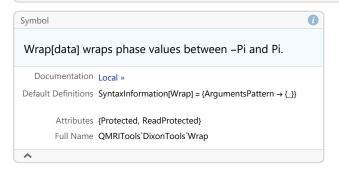




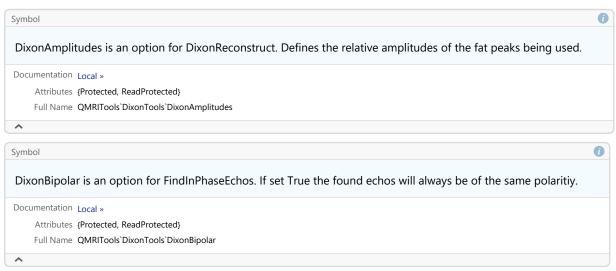


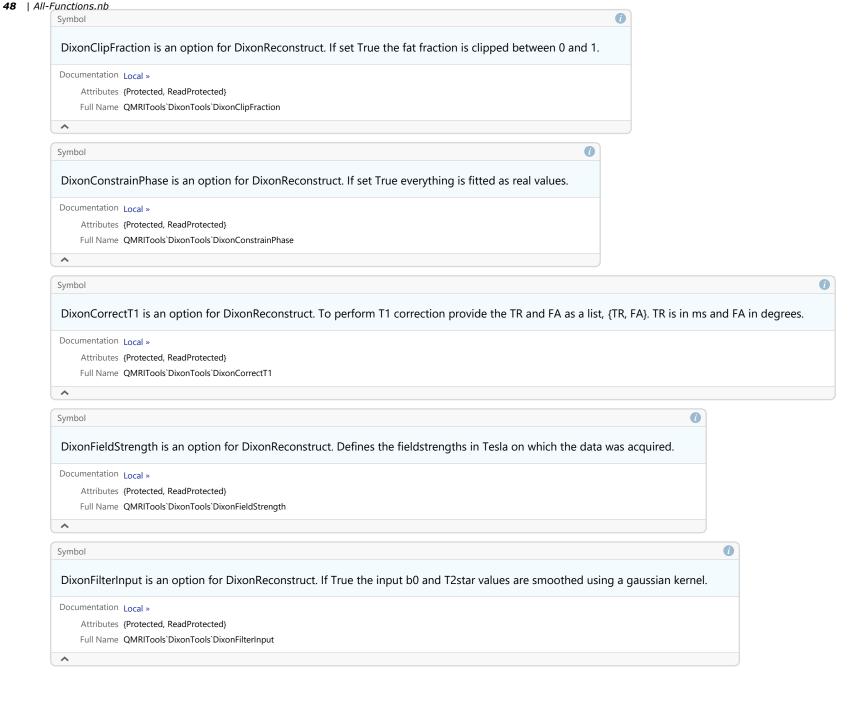
```
Symbol
FindInPhaseEchos[echos, iop] finds the two nearest echos to inphase which are best used for unwrapping using the iop time.
  Documentation Local »
Default Definitions SyntaxInformation[FindInPhaseEchos] = {ArgumentsPattern → {_, _, OptionsPattern[]}}
         Options DixonBipolar → False
       Attributes {Protected, ReadProtected}
       Full Name QMRITools`DixonTools`FindInPhaseEchos
^
                                                                                                                                 0
Symbol
FixDixonFlips[{mag, phase, real, imag}] checks if any volumes are 180 degrees out of phase and corrects them.
  Documentation Local »
Default Definitions SyntaxInformation[FixDixonFlips] = {ArgumentsPattern → {{_, _, _, _}}}}
       Attributes {Protected, ReadProtected}
       Full Name QMRITools'DixonTools'FixDixonFlips
^
Symbol
GenerateAmps[amp] generates the fat based chemical species amplitudes needed for DixonReconstruct.
Documentation Local »
    Attributes {Protected, ReadProtected}
    Full Name QMRITools`DixonTools`GenerateAmps
\wedge
Symbol
OptimizeDixonEcho[] shows a manipulate pannel which allos to optimize the dixon echos.
OptimizeDixonEcho[echos] shows a manipulate pannel which allos to optimize the predifined dixon echos.
Documentation Local »
      Options {DixonNucleus → 1H, DixonFrequencies → {{0,}, {-3.81, -3.4, -3.12, -2.67, -2.45, -1.94, -0.63, -0.4, 0.52, 0.62}}, DixonAmplitudes → {{1}, {0.089, 0.577, 0.059, 0.093, 0.059, 0.013, 0.02, 0.01, 0.059}}}
     Attributes {Protected, ReadProtected}
    Full Name QMRITools`DixonTools`OptimizeDixonEcho
```



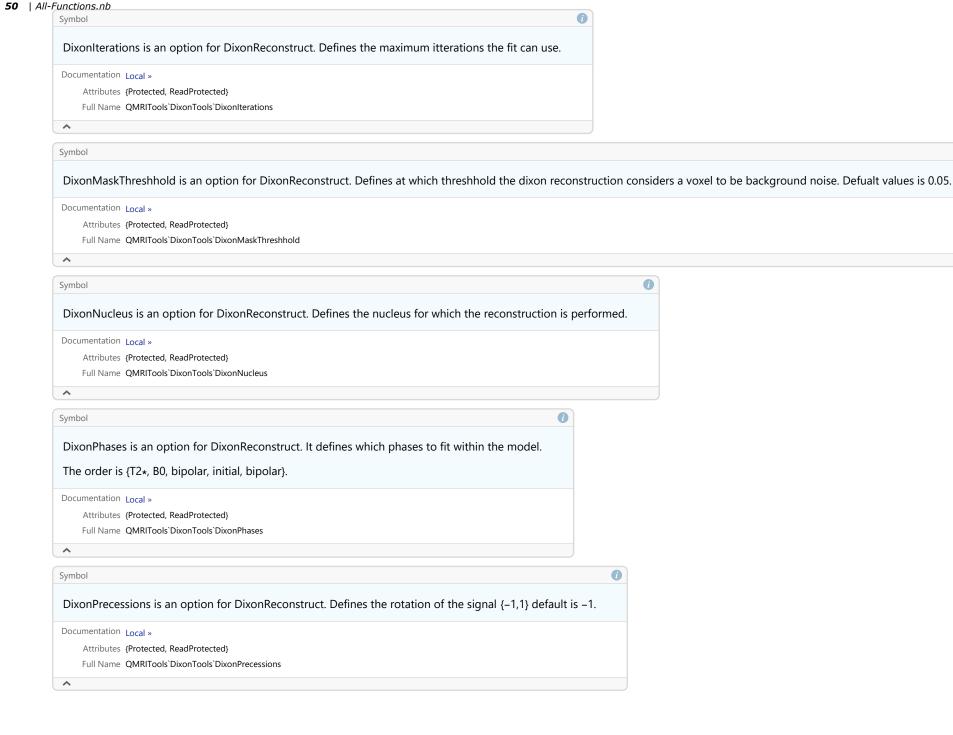


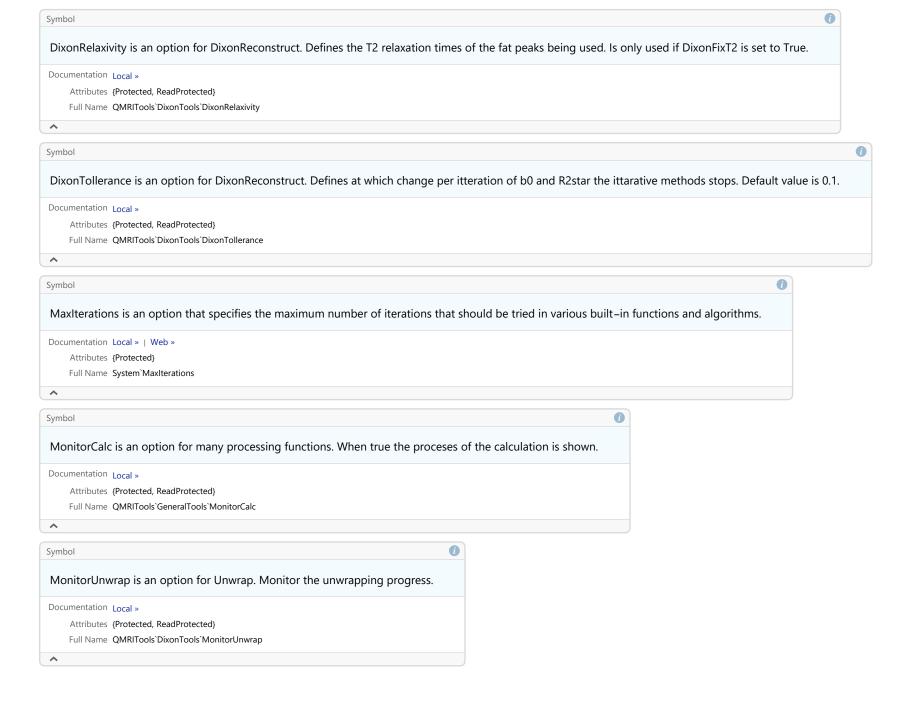
Options

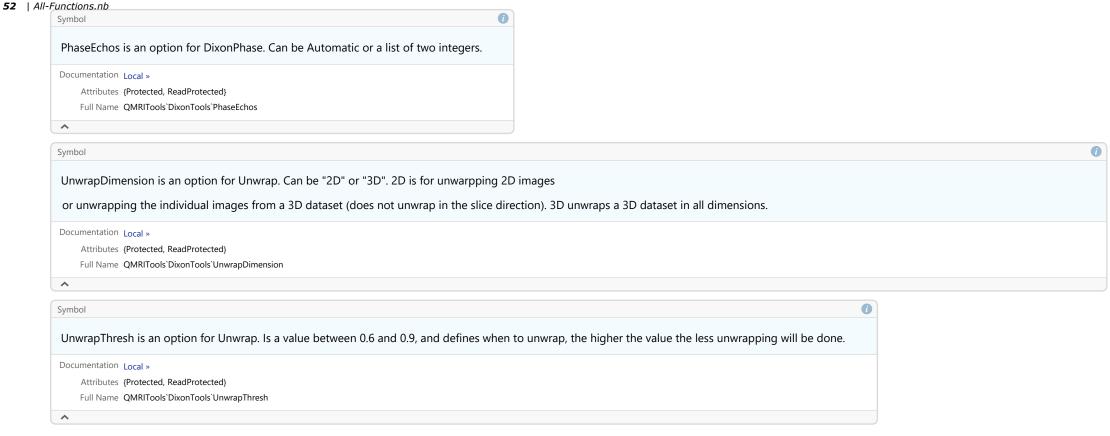




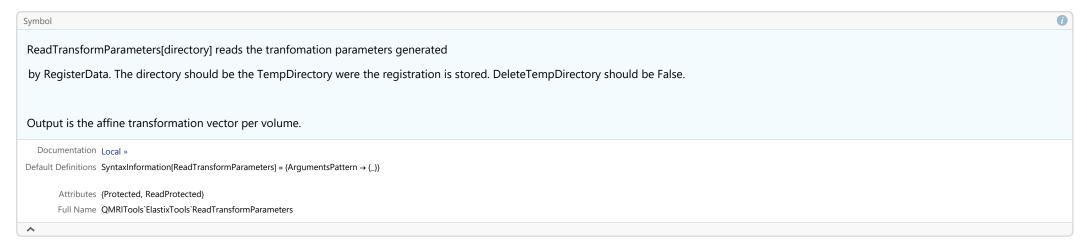
Full Name QMRITools'DixonTools'DixonFrequencies







ElastixTools



```
RegisterCardiacData[data] registers the data using a 2D algorithm. data can be 3D or 4D.

RegisterCardiacData[{data,vox}] registers the data series using the given voxel size.

RegisterCardiacData[{data,mask}] registers the data series only using data whithin the mask.

RegisterCardiacData[{data,mask,vox}] registers the data series using the given voxel size only using data within the mask.

Output is the registered data.

Documentation Local »

Default Definitions SyntaxInformation[RegisterCardiacData] = {ArgumentsPattern → {_, OptionsPattern[]}}

Options ➤ RegistrationTarget → Mean...(18 total)

Attributes {Protected, ReadProtected}

Full Name QMRITools'ElastixTools'RegisterCardiacData
```

Symbol

RegisterData[data] registers the data series. If data is 3D it performs multiple 2D registration, if data is 4D it performs multiple 3D registration.

The input data can be in the forms: data, {data, vox}, {data, mask} or {data, mask, vox}.

RegisterData[target, moving] registers the moving data to the target data. target can be 2D or 3D. moving can be the same of one dimension higher than the target.

The inputs target and moving can be in the forms: data, {data, vox}, {data, mask} or {data, mask, vox}.

Output is the registered data with the dimensions of the moving data.

If OutputTransformation is True it also outputs the translation, rotation scale and skew of all images or volumes.

RegisterData[] is based on DOI: 10.1109/TMI.2009.2035616 and 10.3389/fninf.2013.00050.

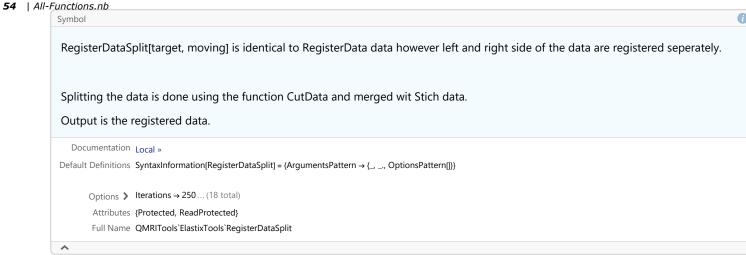
Documentation Local »

Default Definitions SyntaxInformation[RegisterData] = {ArgumentsPattern → {_, _, OptionsPattern[]}}

Options ➤ Iterations → 250 ... (17 total)

Attributes {Protected, ReadProtected}

Full Name QMRITools`ElastixTools`RegisterData



RegisterDataTransform[target, moving, {moving2nd, vox}] performs the registration exactly as RegisterData. target and moving are the inputs for Registerdata, which can be {data,mask,vox}.

After the registeration is done the moving2nd data is deformed acording to the output of the registrition of moving.

moving2nd can have the same dimensions of moving or one dimension higher (e.g. 3D and 3D or 3D and 4D).

Output is {registered moving, deformed moving2nd}.

Documentation Local >

Default Definitions SyntaxInformation[RegisterDataTransform] = {ArgumentsPattern → {. OptionsPattern[]}}

Options > Iterations → 250...(18 total)

Attributes (Protected, ReadProtected)

Full Name QMRITools ElastixTools RegisterDataTransform

```
RegisterDataTransformSplit[target, moving, {moving2nd, vox}] is idenditcal to RegisterDataTransform with the same functionality as RegisterDataSplit.

This means the data is split in two using the function CutData and merged wit Stich data.

Output is {registered moving, deformed moving2nd}.

Documentation Local >>
Default Definitions SyntaxInformation[RegisterDataTransformSplit] = {ArgumentsPattern → { __ _ _ _ _ OptionsPattern[]}}

Options >> Iterations → 250 ... (19 total)
Attributes {Protected, ReadProtected}
Full Name QMRITools'ElastixTools'RegisterDataTransformSplit
```

Symbol

 \wedge

RegisterDiffusionData[{dtidata, vox}] registers a diffusion dataset. dtidata should be 4D {slice, diff, x, y}. vox is the voxelsize of the data.

RegisterDiffusionData[{dtidata, dtimask, vox}] registers the data series using the given voxel size only using data within the mask.

RegisterDiffusionData[{dtidata,vox}, {anatdata, voxa}] registers a diffusion dataset. The diffusion data is also registered to the anatdata.

RegisterDiffusionData[{dtidata, dtimask, vox}, {anatdata, voxa}] registers the data series using the given voxel size only using data within the mask.

RegisterDiffusionData[{dtidata,vox}, {anatdata, anatmask, voxa}] registers the data series using the given voxel size only using data within the mask.

RegisterDiffusionData[{dtidata, dtimask, vox}, {anatdata, anatmask, voxa}] registers the data series using the given voxel size only using data within the mask.

Output is the registered dtidata and, if anatdata is given, the registered dtidata in

anatomical space. If OutputTransformation is True it also outputs the translation, rotation scale and skew of all images or volumes.

```
Documentation Local »

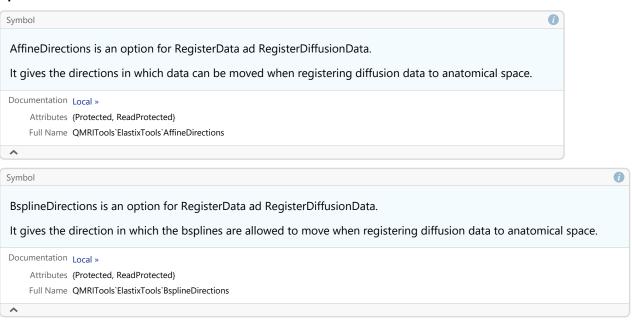
Default Definitions SyntaxInformation[RegisterDiffusionData] = {ArgumentsPattern → {_, _, _, OptionsPattern[]}}

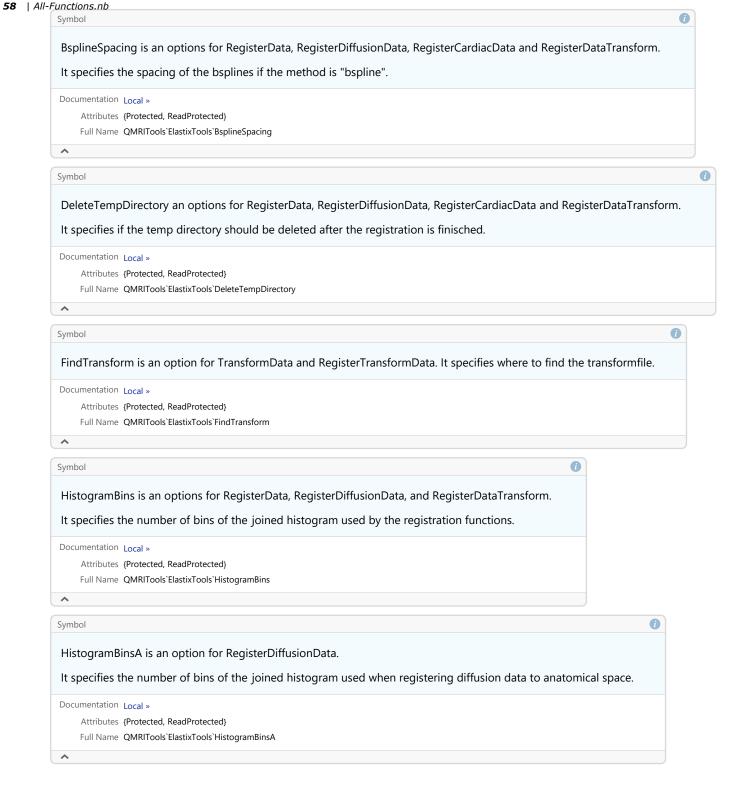
Options > Iterations → 250... (24 total)

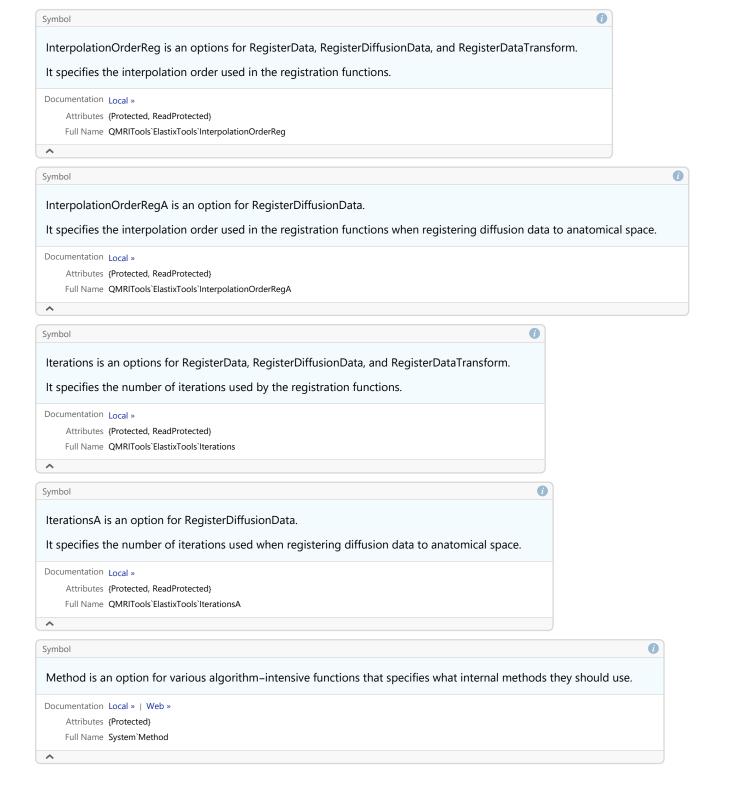
Attributes {Protected, ReadProtected}

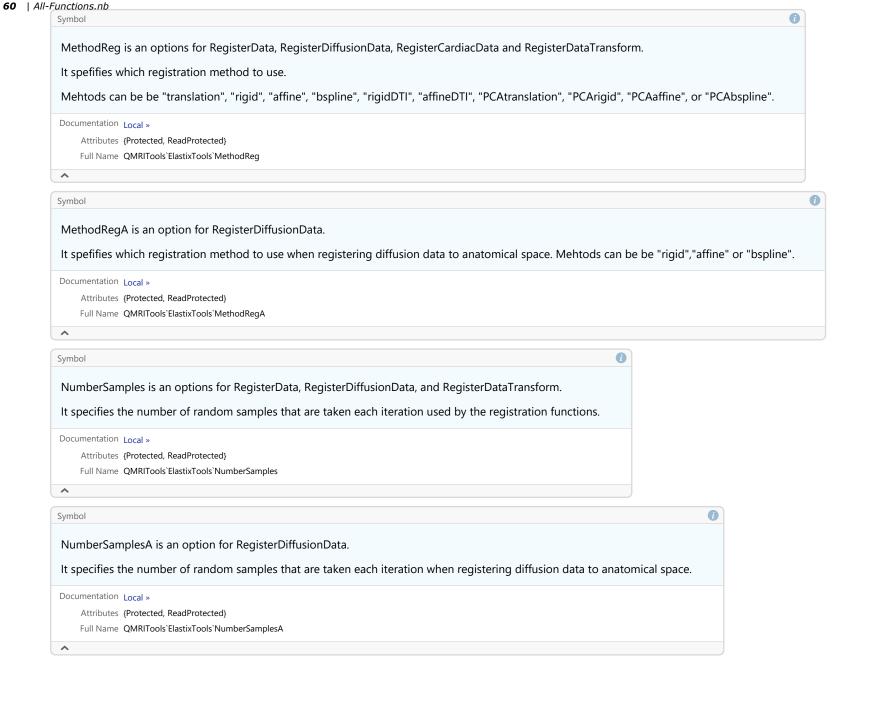
Full Name QMRITools'ElastixTools'RegisterDiffusionData
```

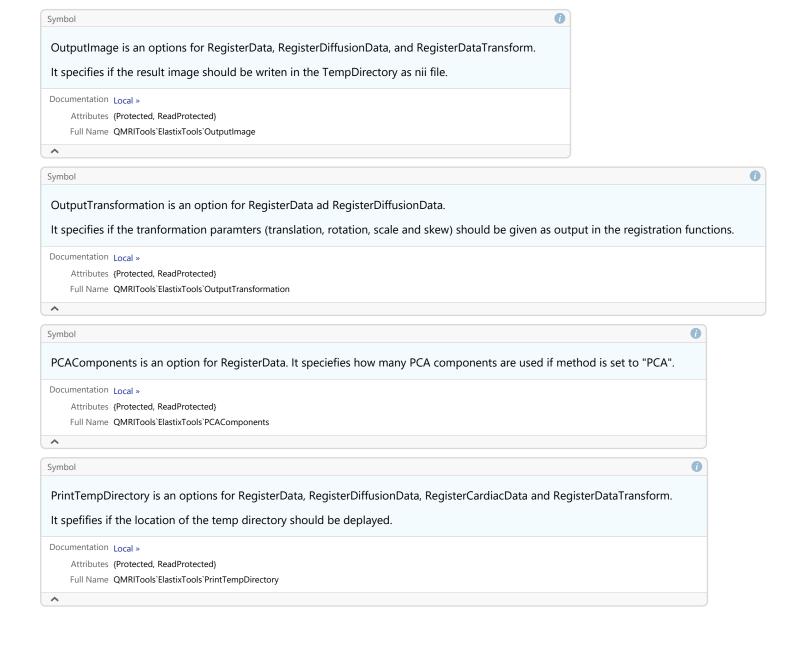
```
Symbol
  TransformData[{data,vox}] deforms the data according to the last output of register data.
  The directory should be the TempDirectory were the registration is stored. DeleteTempDirectory should be False.
    Documentation Local »
  Default\ Definitions\ \ SyntaxInformation[TransformData] = \{ArgumentsPattern \rightarrow \{\_,\ OptionsPattern[]\}\}
        Options ➤ TempDirectory → Default ... (5 total)
         Attributes {Protected, ReadProtected}
         Full Name QMRITools`ElastixTools`TransformData
 Symbol
  $debugElastix is a parameter that allows to print Elastix commands if set to True.
  Documentation Local »
      Attributes {ReadProtected}
      Full Name QMRITools`ElastixTools`$debugElastix
Options
                                                                                                                             0
 Symbol
```

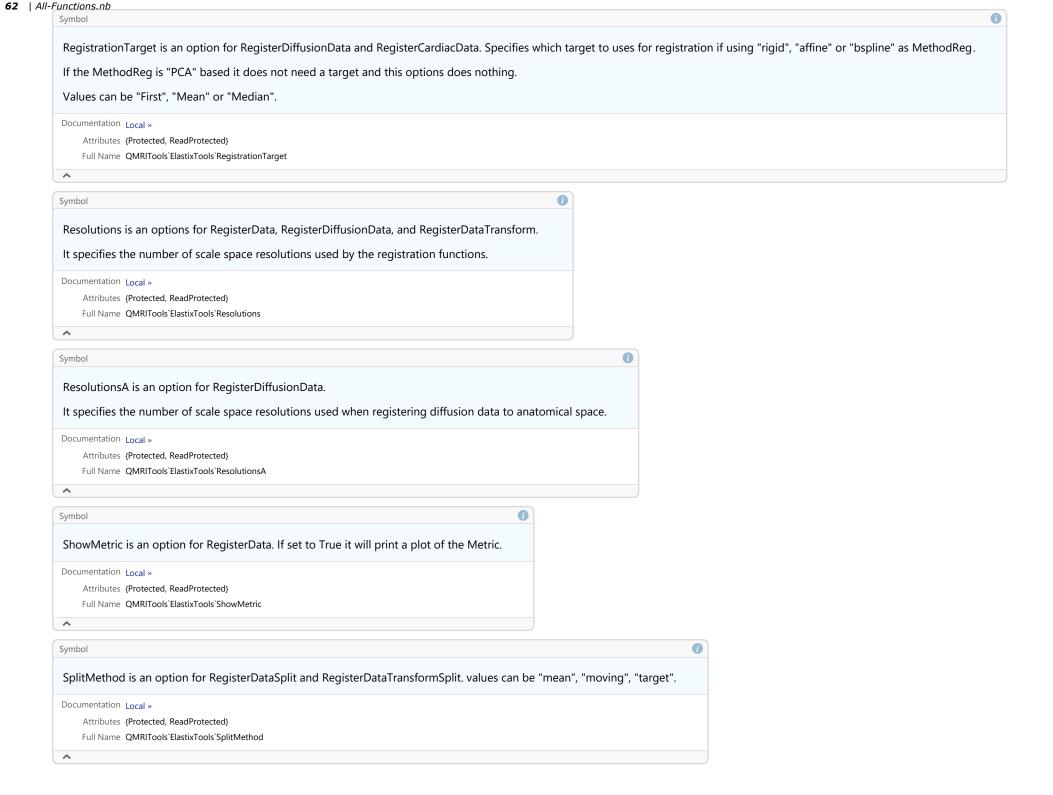


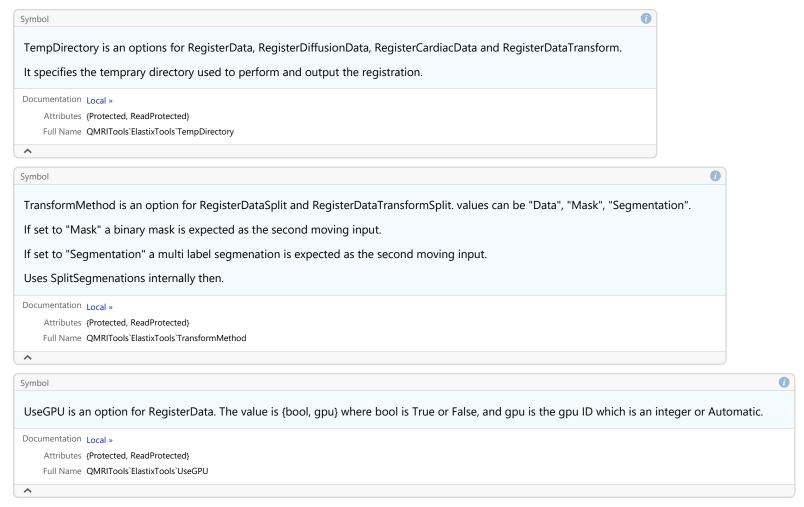




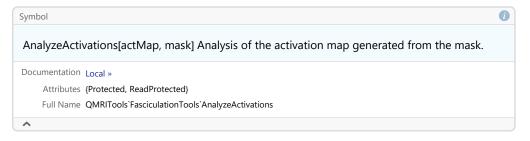


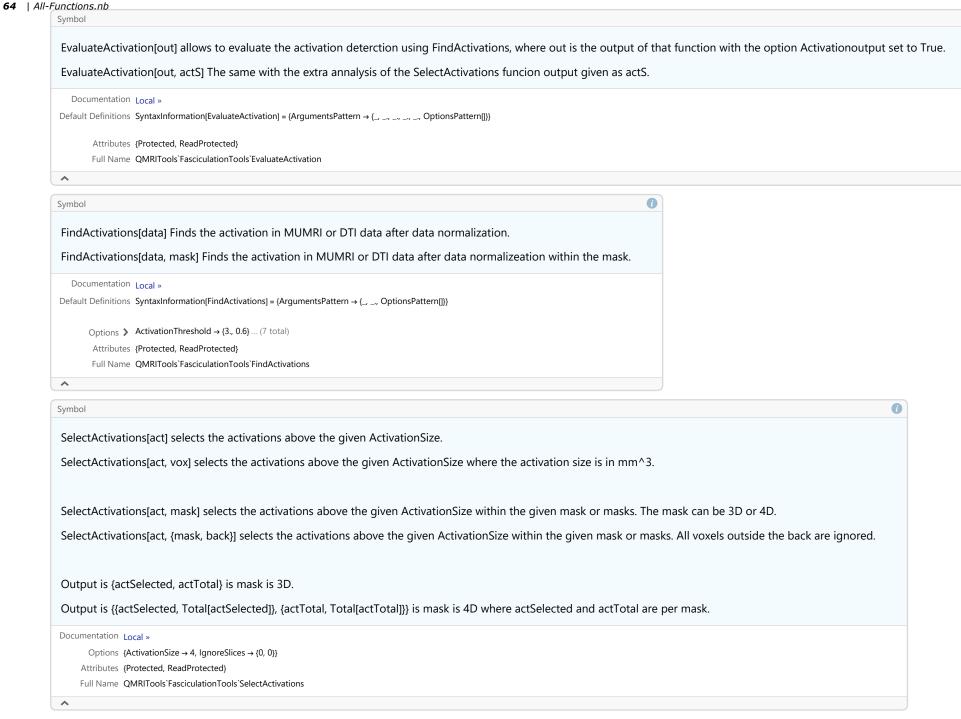




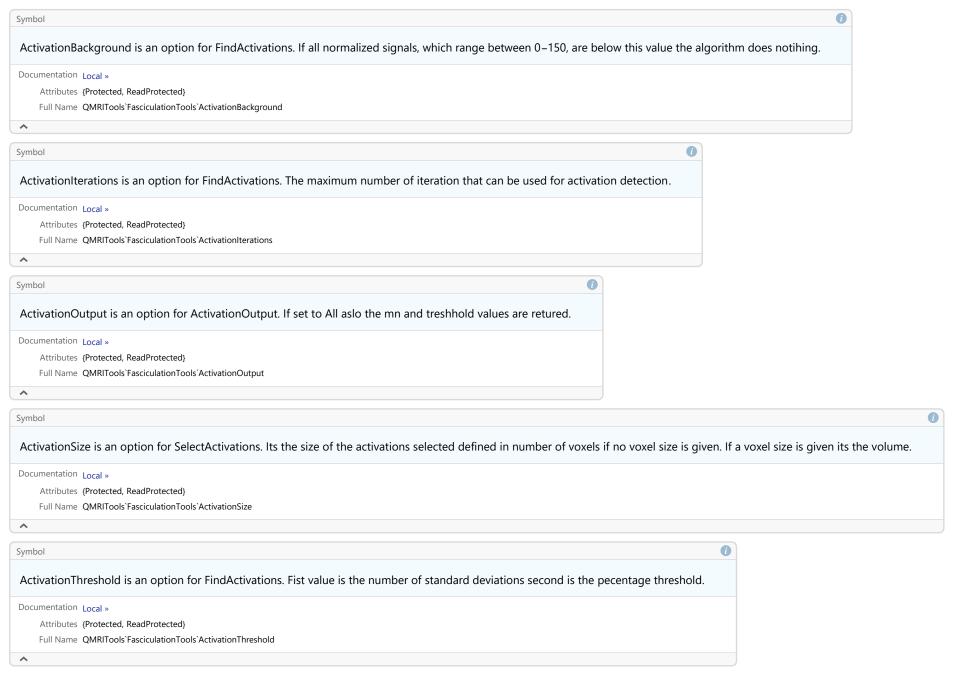


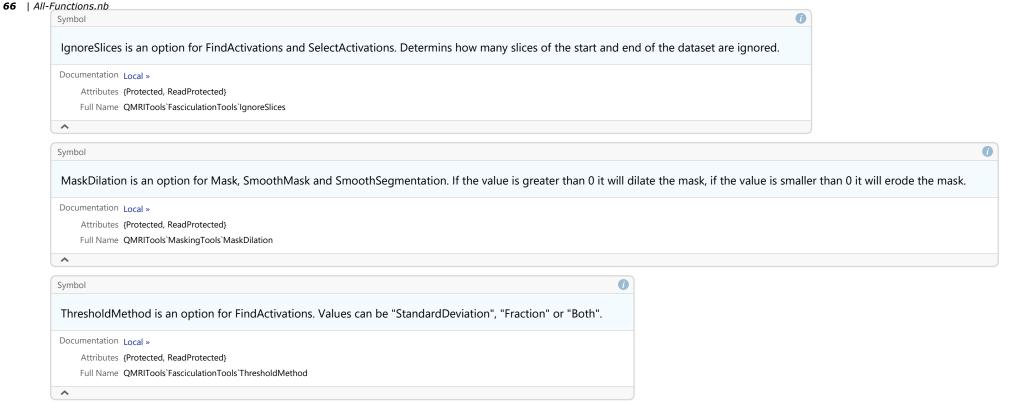
FasciculationTools



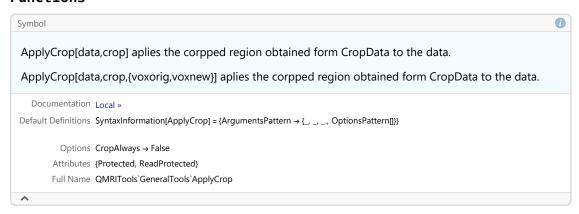


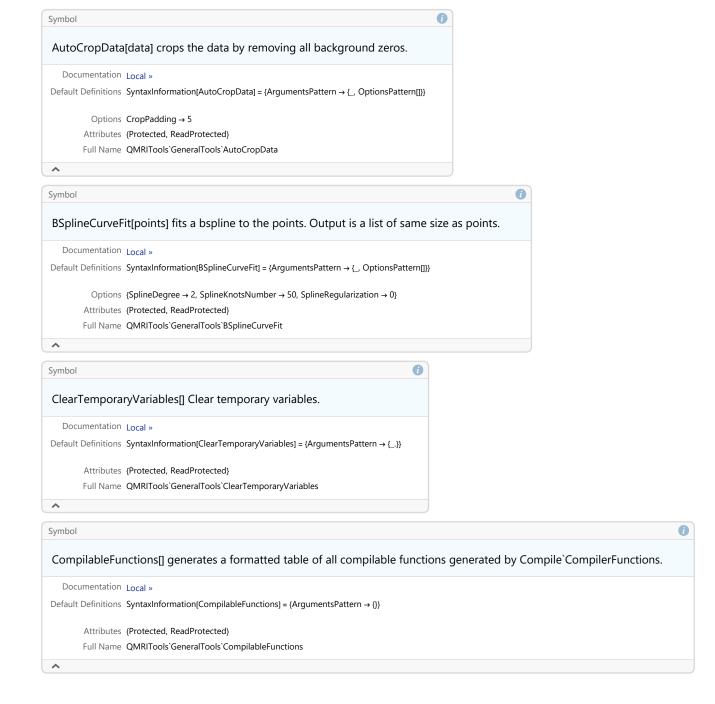
Options

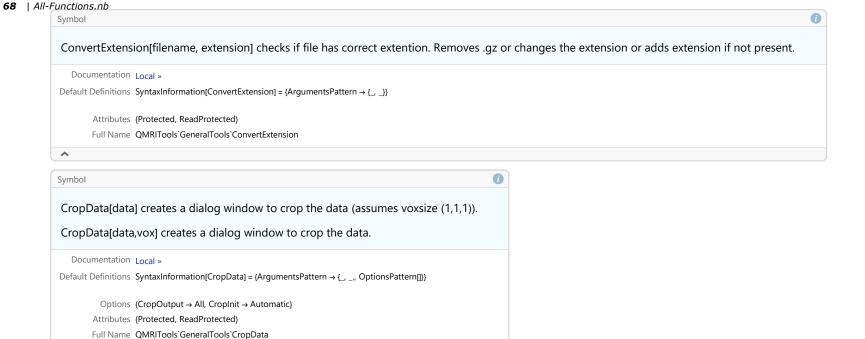


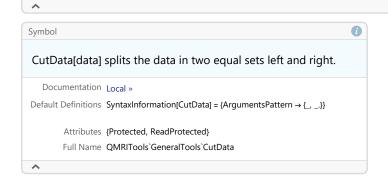


GeneralTools

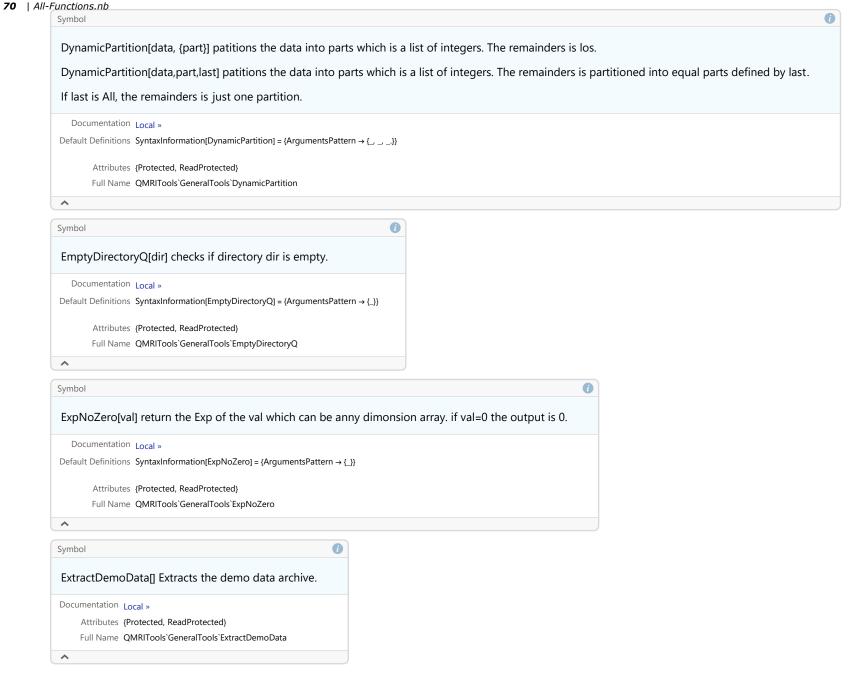


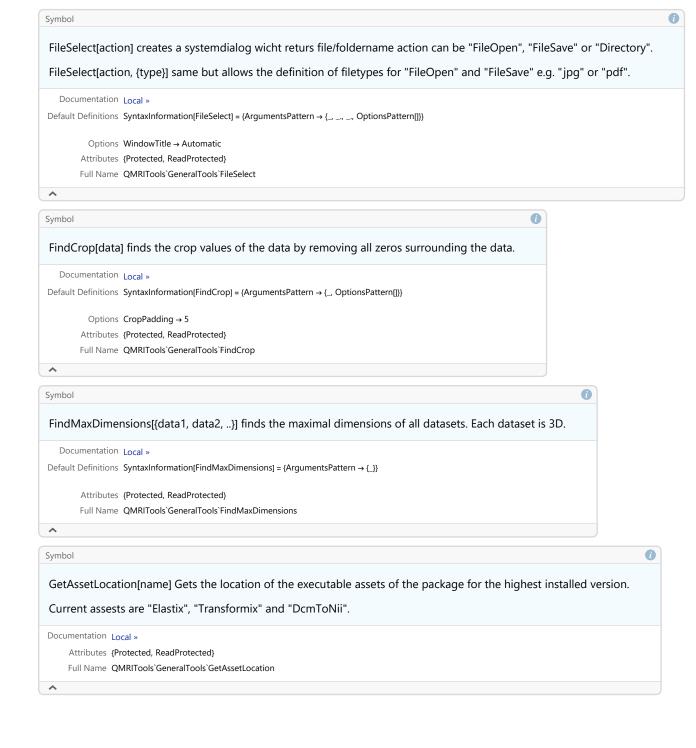


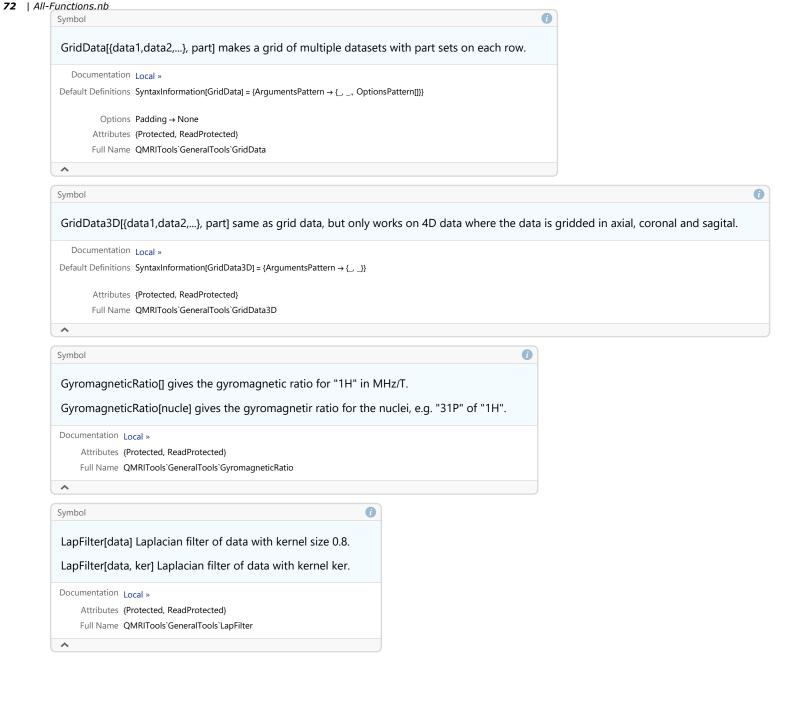


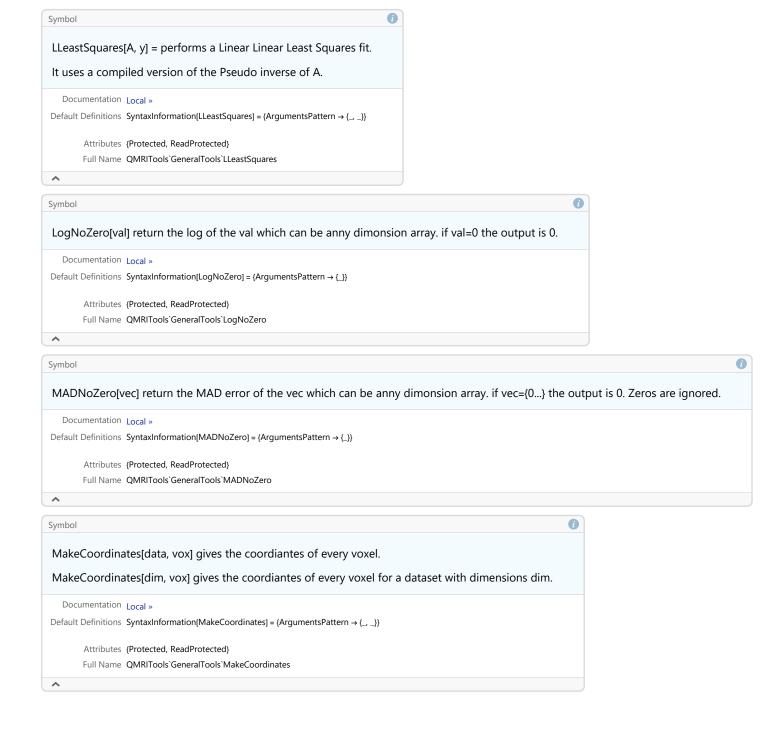


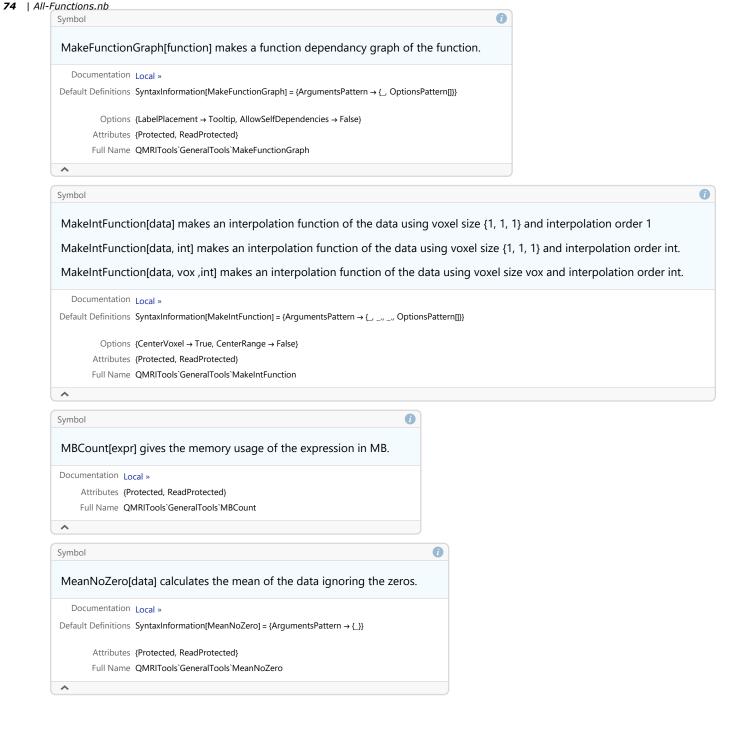
```
Symbol
DataToVector[data] converst the non zero data to vector.
DataToVector[data, mask] converst the data within the mask to vector.
the data can be reconstructed using VectorToData.
output is the vecotrized data and a list contining the original data dimensions and a list with the data coordinates. {vec, {dim,pos}}.
  Documentation Local »
Default Definitions SyntaxInformation[DataToVector] = {ArgumentsPattern → {_, _.}}
       Attributes {Protected, ReadProtected}
       Full Name QMRITools`GeneralTools`DataToVector
^
                                                                                        0
Symbol
DecomposeAffineMatrix[S] decomposes the scale matrix in S1, S2 and S3.
  Documentation Local »
Default Definitions SyntaxInformation[DecomposeAffineMatrix] = \{ArgumentsPattern \rightarrow \{\_\}\}
       Attributes {Protected, ReadProtected}
       Full Name QMRITools'GeneralTools'DecomposeAffineMatrix
^
Symbol
DecomposeScaleMatrix[mat] decomposes the affine matirx in T, R, S and Q.
  Documentation Local »
Default Definitions SyntaxInformation[DecomposeScaleMatrix] = {ArgumentsPattern \rightarrow {_}}
       Attributes {Protected, ReadProtected}
       Full Name QMRITools`GeneralTools`DecomposeScaleMatrix
^
Symbol
DevideNoZero[a, b] devides a/b but when b=0 the result is 0. a can be a number or vector.
  Documentation Local »
Default Definitions SyntaxInformation[DevideNoZero] = {ArgumentsPattern \rightarrow {_, _, _, _.}}
       Attributes {Protected, ReadProtected}
       Full Name QMRITools`GeneralTools`DevideNoZero
\wedge
```

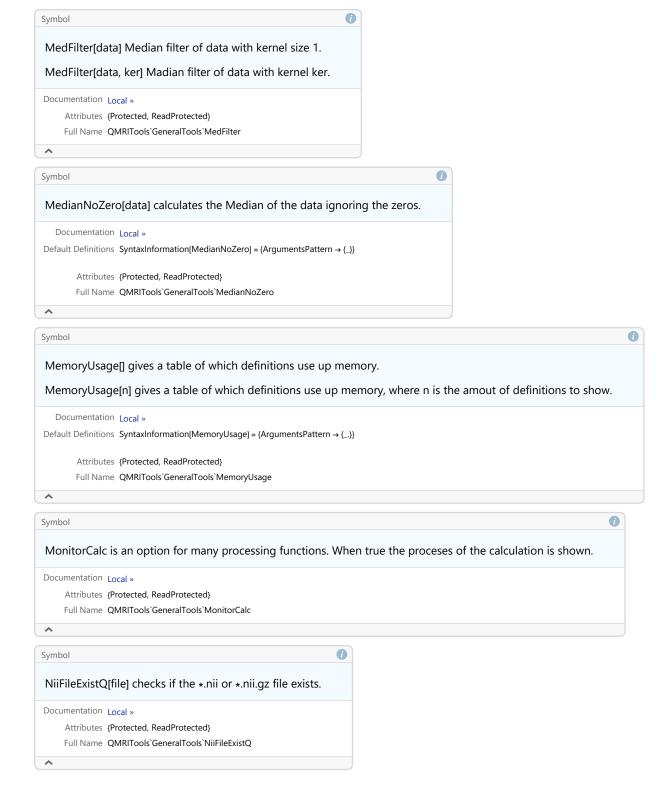


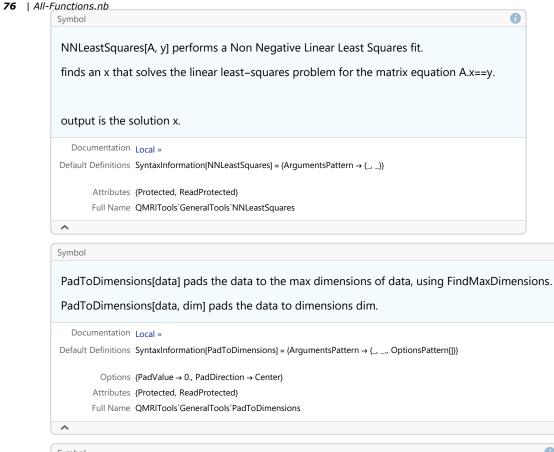


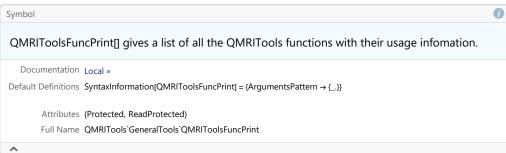


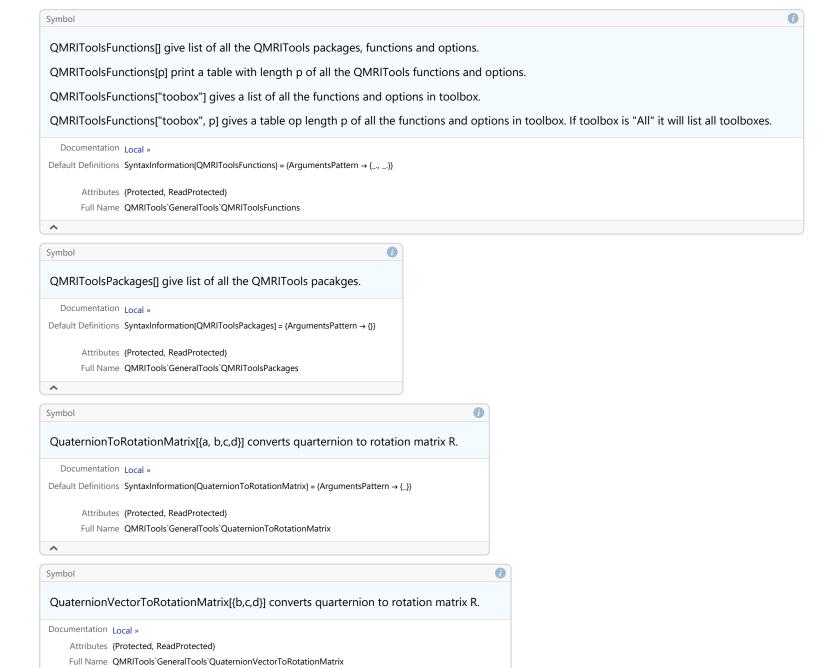


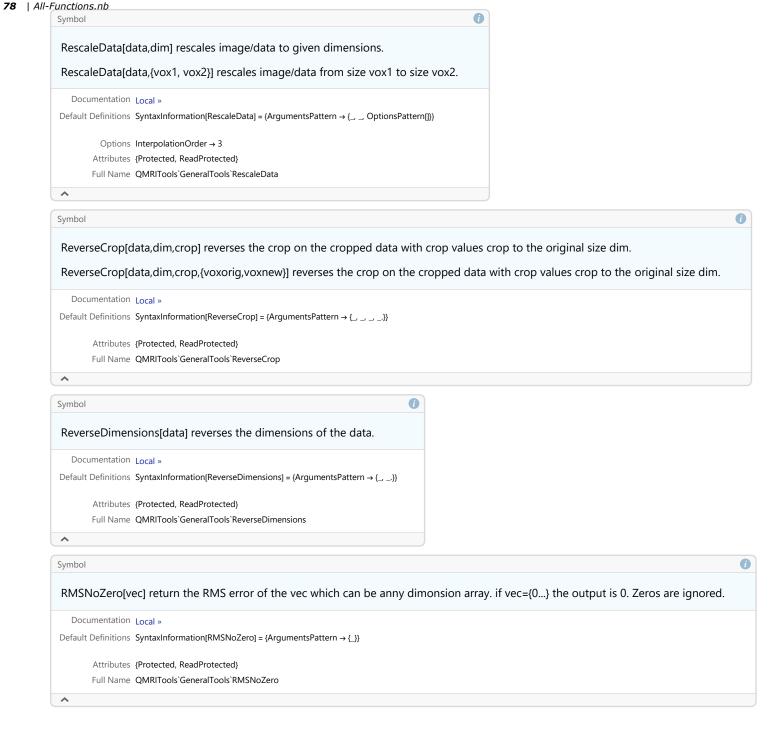




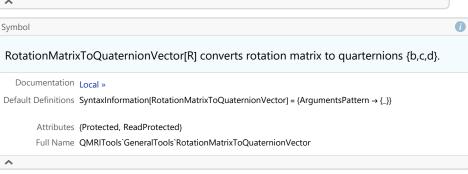


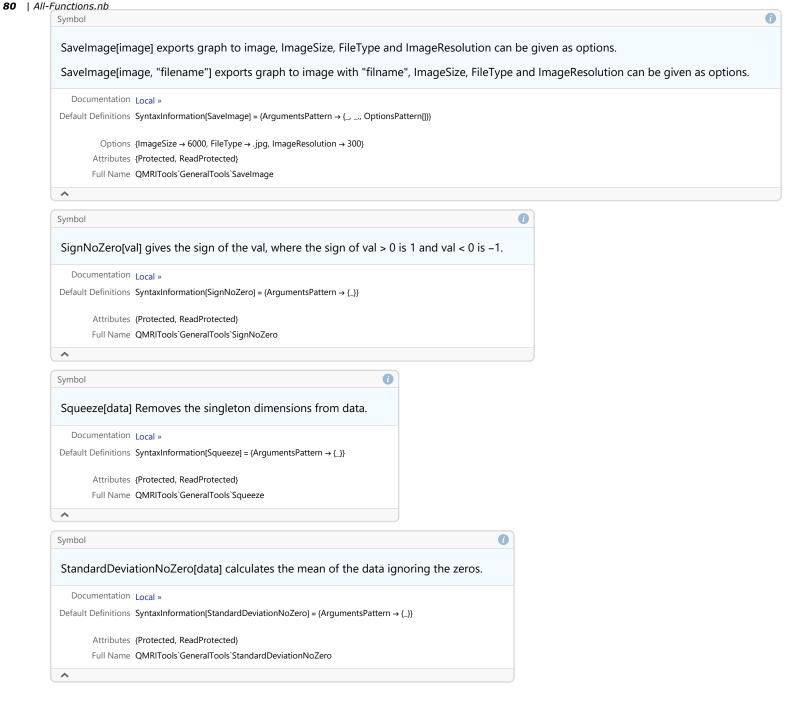


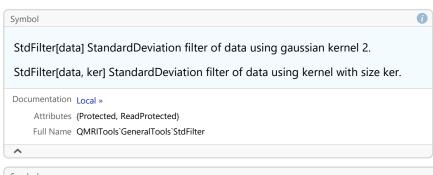


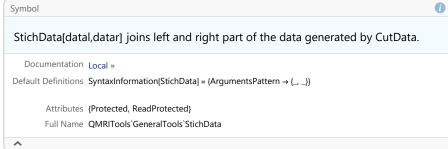


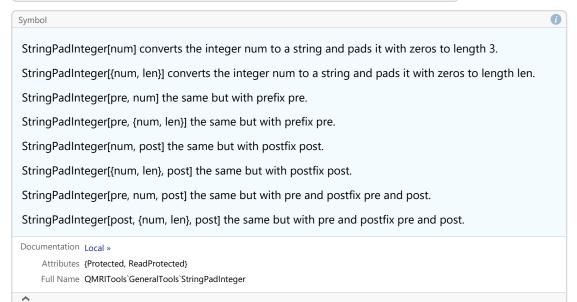
All-Functions.nb | 79

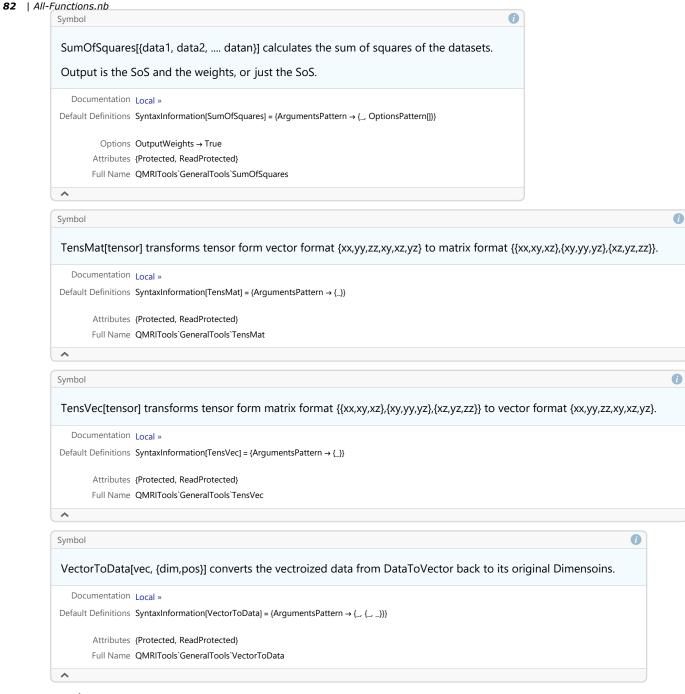




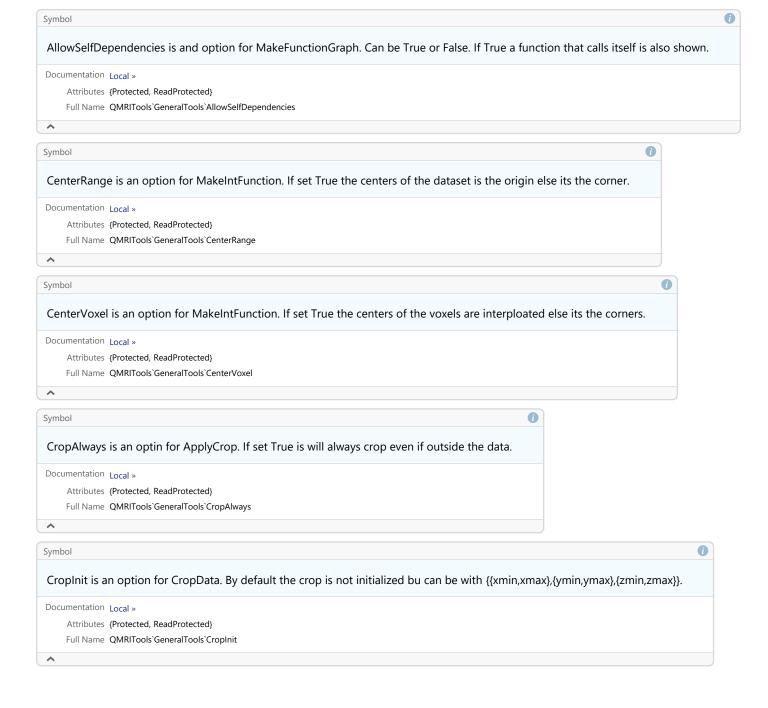


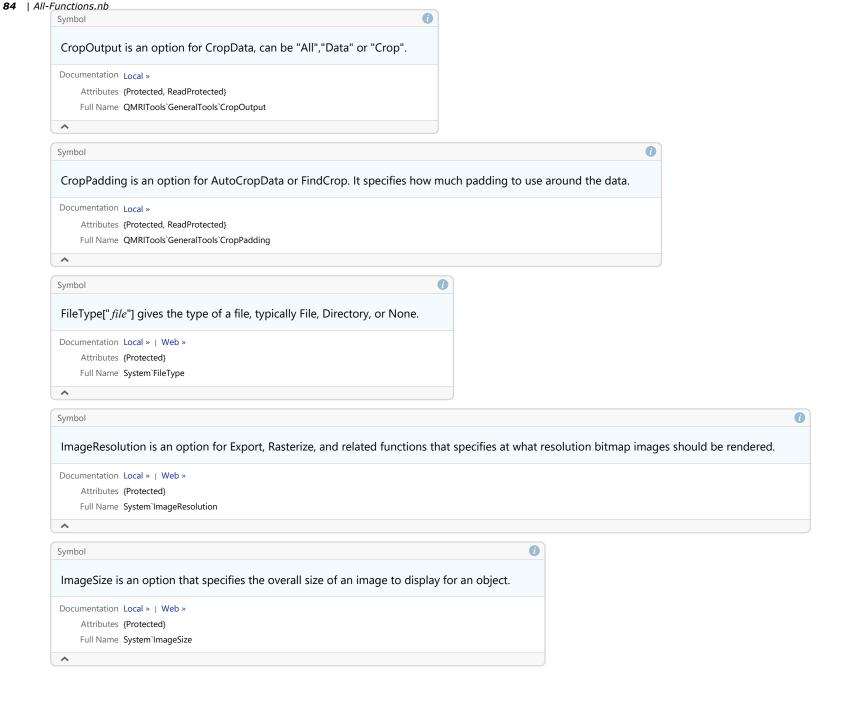


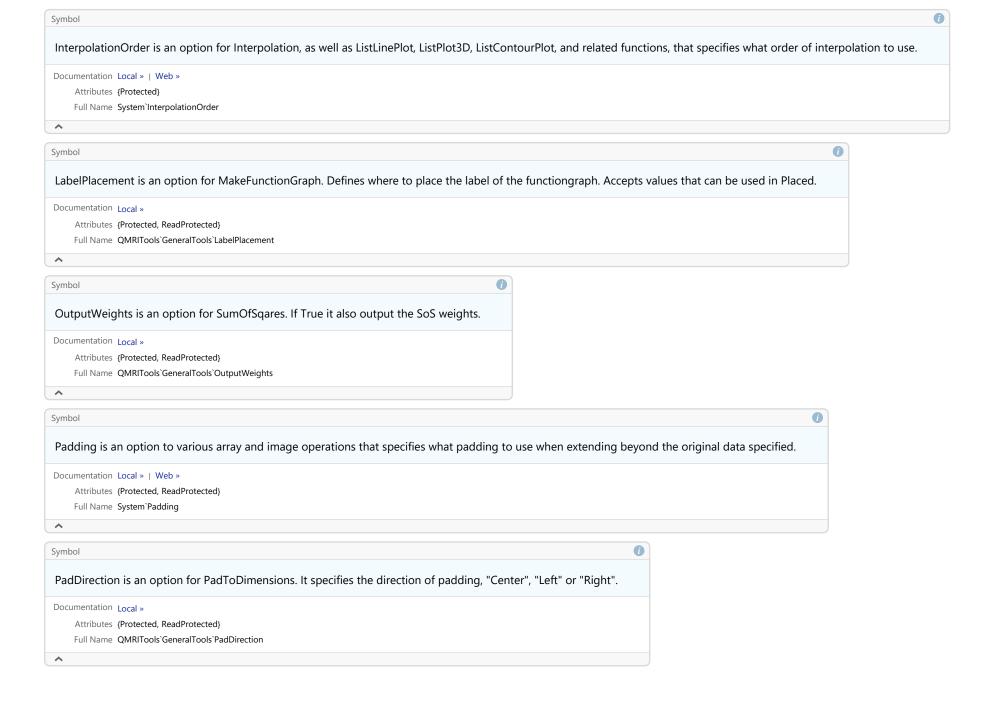


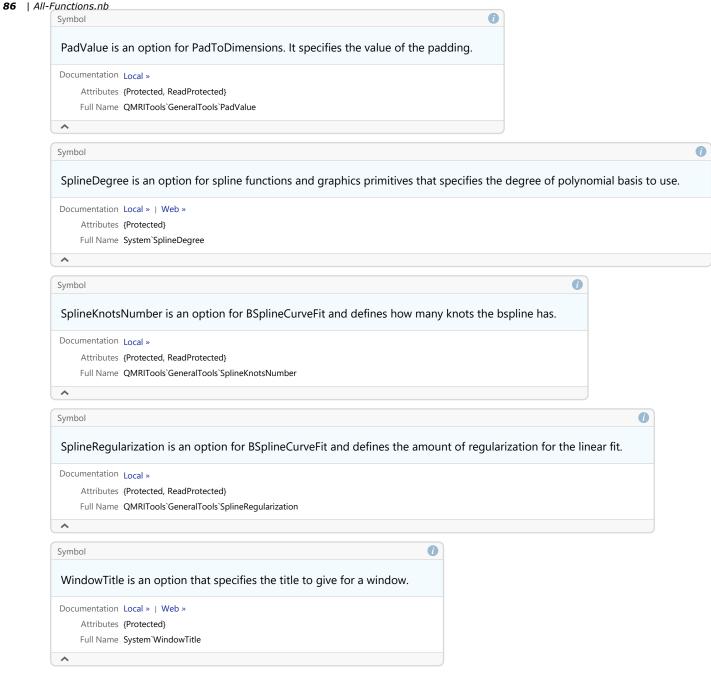


Options





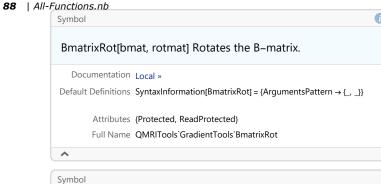


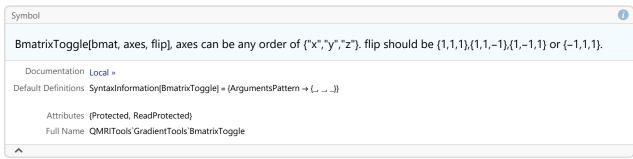


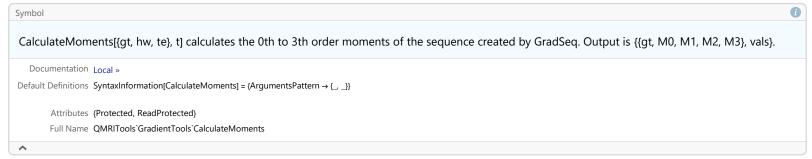
GradientTools

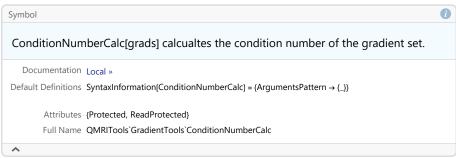
Functions

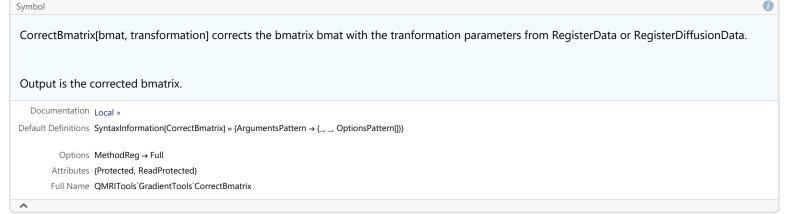
```
Symbol
Bmatrix[bvec, grad] creates bmatrix form grad and bvec in form {-bxx, -byy, -bzz, -bxy, -bxz, -byz, -1}.
Bmatrix[{bvec, grad}] creates bmatrix form grad and bvec in form {bxx, byy, bzz, bxy, bxz, byz}.
   Documentation Local »
Default Definitions SyntaxInformation[Bmatrix] = {ArgumentsPattern → {_, _,, OptionsPattern[]}}
          Options Method → DTI
        Attributes {Protected, ReadProtected}
       Full Name QMRITools`GradientTools`Bmatrix
^
Symbol
BmatrixCalc["folder", grads] calculates the true bmatrix from the exported sequence parameters from the philips scanner that are stored in "folder" for each of the gradient directions grads.
   Documentation Local »
Default Definitions SyntaxInformation[BmatrixCalc] = {ArgumentsPattern → {_, _, _, _, OptionsPattern[]}}
       Options > UseGrad → {1, 1, {1, 1}, 1, 1} ... (8 total)
        Attributes {Protected, ReadProtected}
        Full Name QMRITools`GradientTools`BmatrixCalc
\wedge
                                                                                   0
Symbol
BmatrixConv[bm] converts the bmatrix form 7 to 6 or from 6 to 7.
   Documentation Local »
\label{eq:definitions} Default\ Definitions\ \ SyntaxInformation[BmatrixConv] = \{ArgumentsPattern \rightarrow \{\_\}\}
        Attributes (Protected, ReadProtected)
        Full Name QMRITools`GradientTools`BmatrixConv
\wedge
                                                                                                                                              0
Symbol
BmatrixInv[bm] generates a byecotr and gradiens directions form a given bmatrx.
BmatrixInv[bm, bvi] generates a bvecotr and gradiens directions form a given bmatrx using the given bvalues bvi.
   Documentation Local »
\label{eq:definitions} Default\ Definitions\ \ SyntaxInformation[BmatrixInv] = \{ArgumentsPattern \rightarrow \{\_,\_.\}\}
        Attributes (Protected, ReadProtected)
        Full Name QMRITools`GradientTools`BmatrixInv
\wedge
```

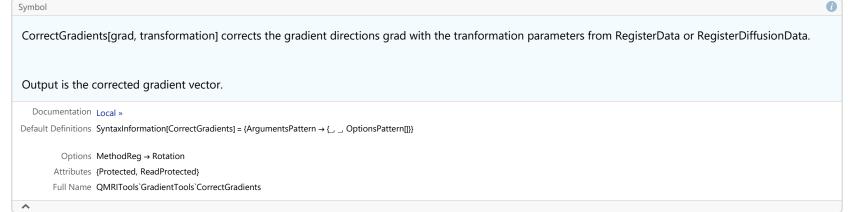


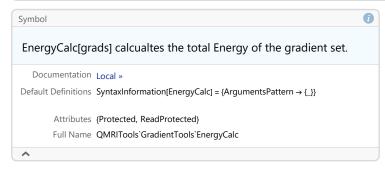


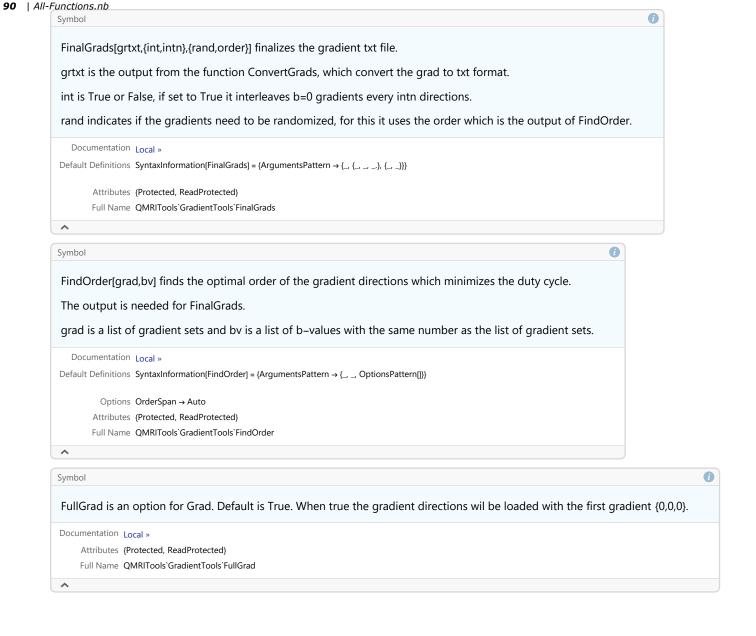




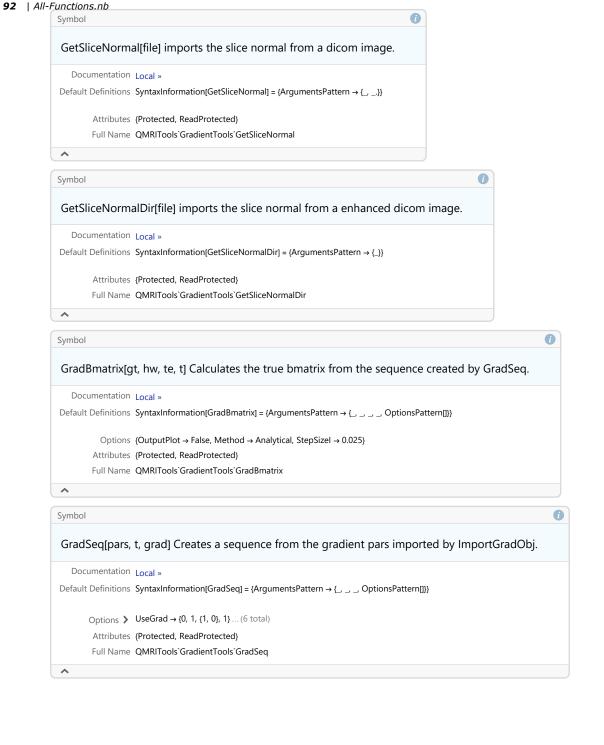


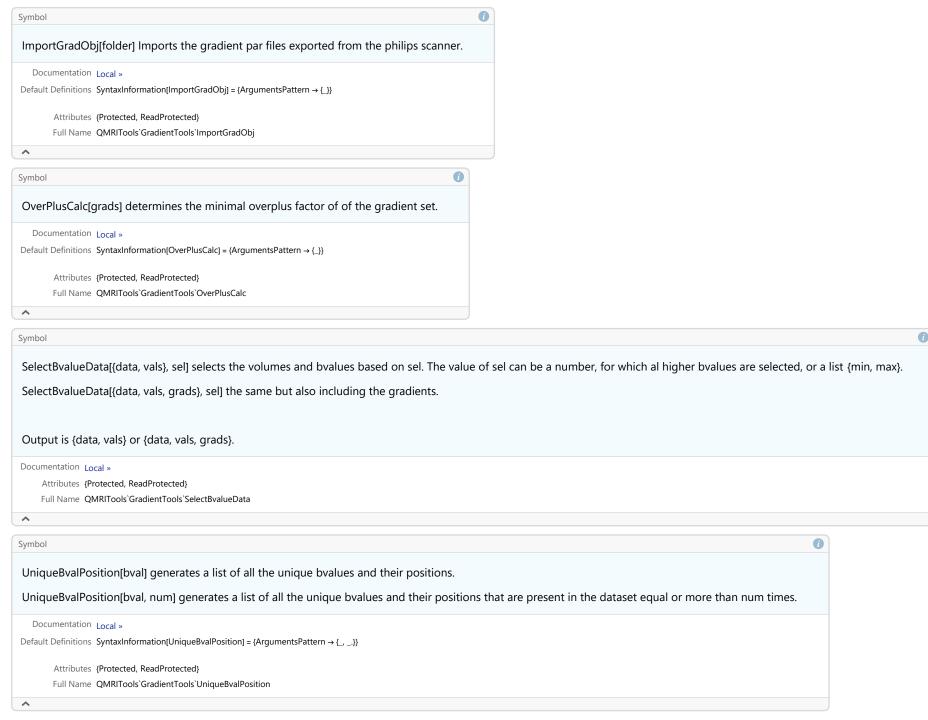




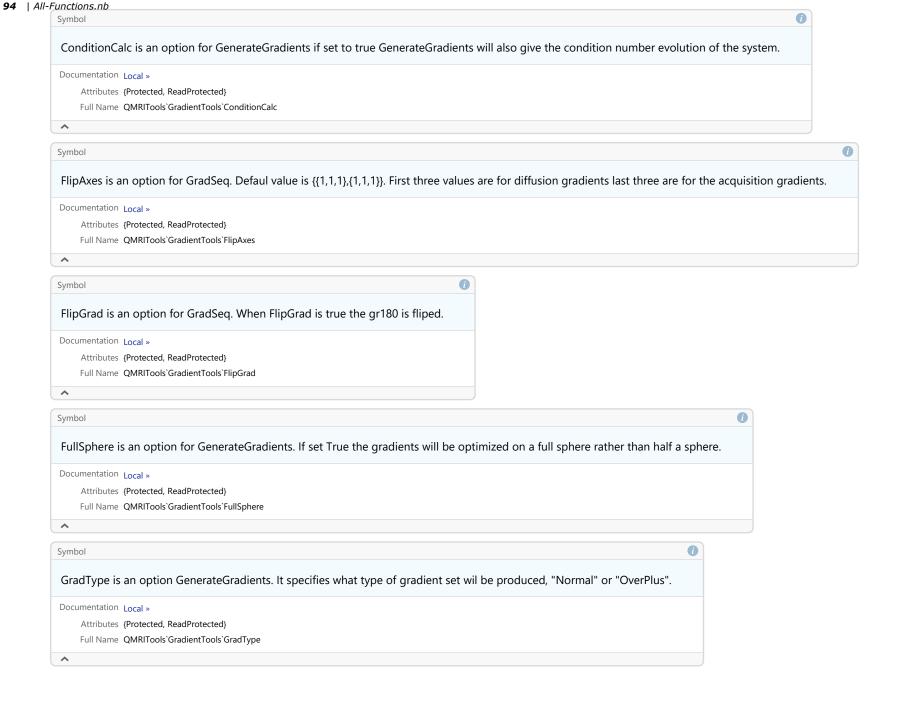


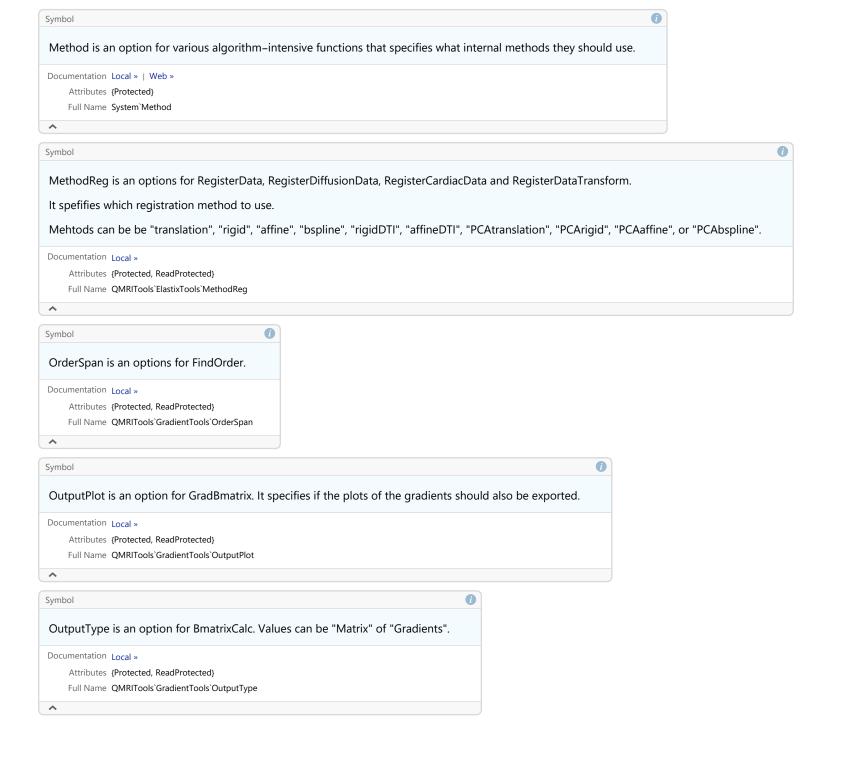
^

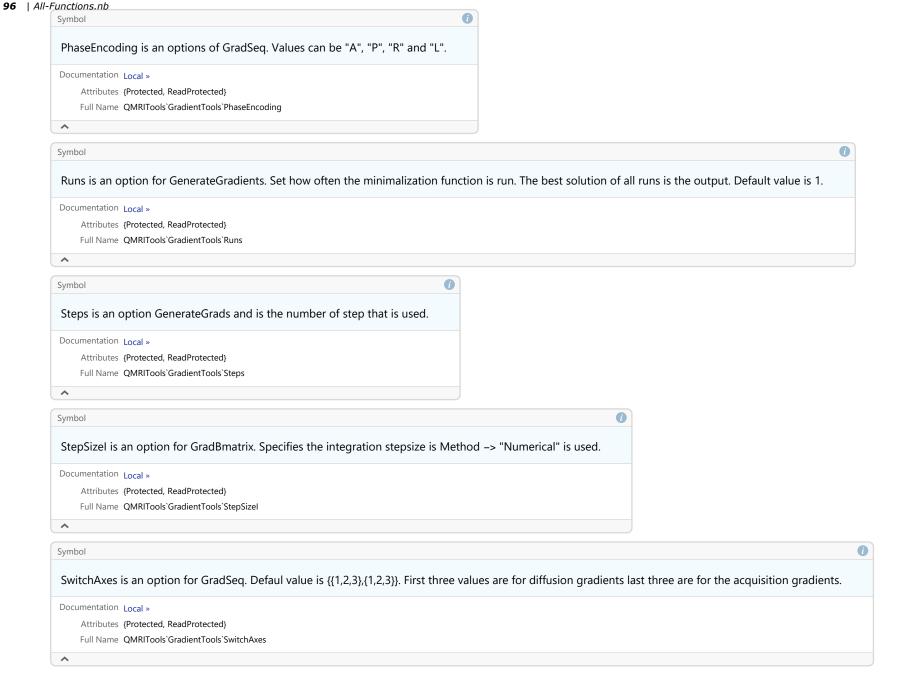


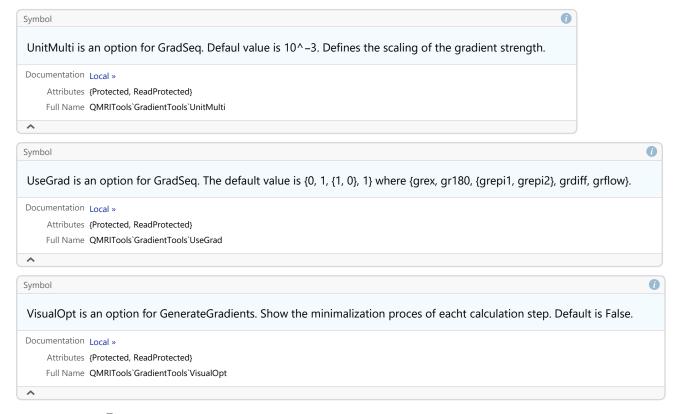


Options



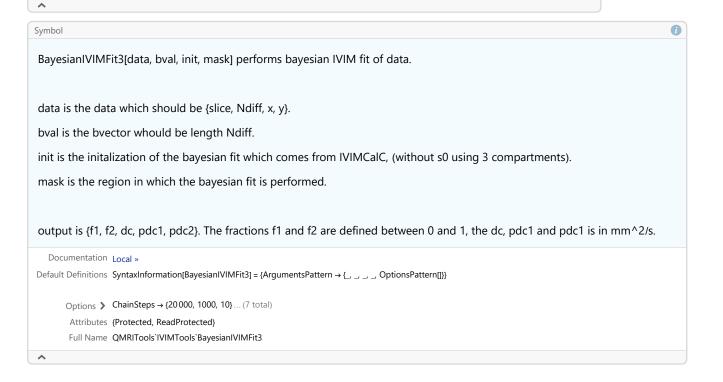


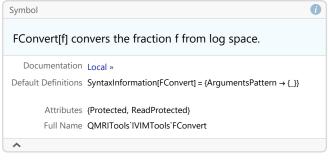


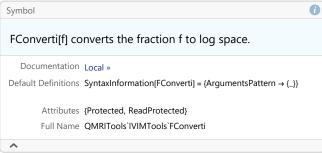


IVIMTools

Functions







All-Functions.nb | 99

100 | All-Functions.nb Symbol After correction the signal fraction can be regarded as volume fraction.

FracCorrect[fraction, time] corrects the signal fraction calculated with the IVIM model for tissue relaxation and acquisition parameters.

0

FracCorrect[{fraction1, fraction2}, time] corrects the signal fraction1 and fraction2 from a 3 compartement IVIM model.

time is {{te, tr}, {t2t, t21}, {t1t, t11}} or {{te, tr}, {t2t, t21, t22}, {t1t, t11, t12}}.

where t2t and t1t are "tissue" relaxation times and t11 t12, t21 and t22 the "fluid" relaxation times.

The te and tr as well as the relaxation times T2 and T1 can be defines in any time unit as long as they are consistant for all, e.g. all in ms.

output is the corrected fraction maps.

Documentation Local »

^

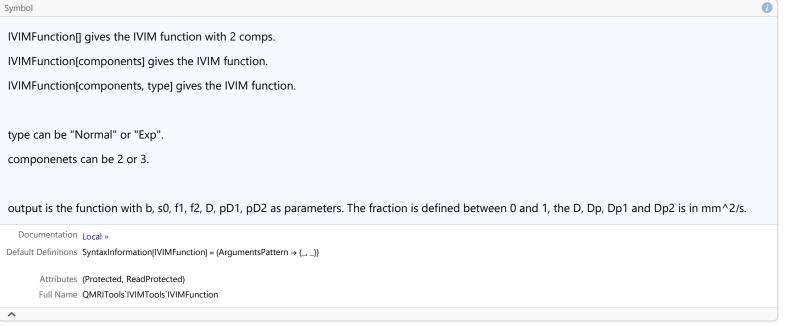
 $Default \ Definitions \ \ SyntaxInformation[FracCorrect] = \{ArgumentsPattern \rightarrow \{_,_,_\}\}$

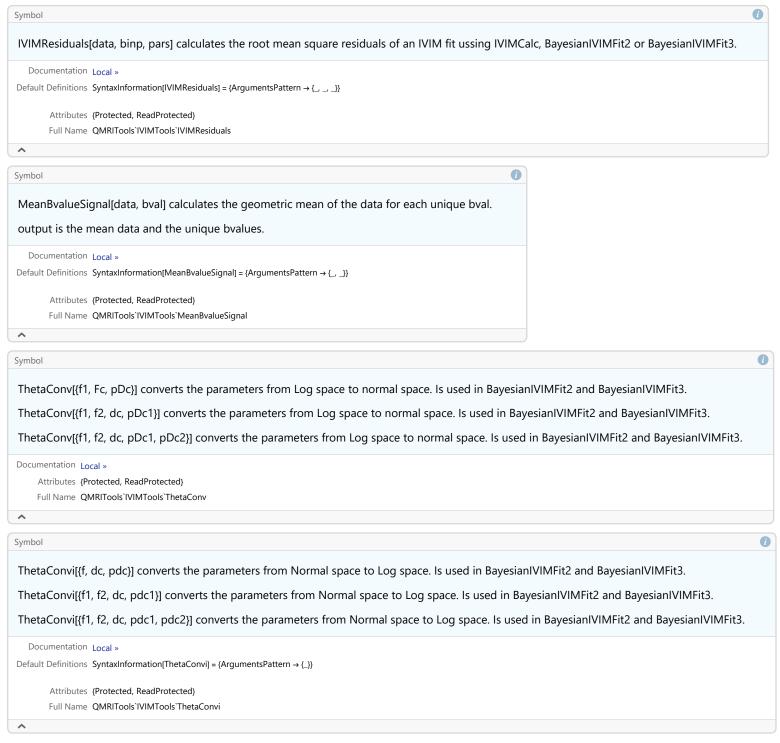
Attributes {Protected, ReadProtected}

Full Name QMRITools`IVIMTools`FracCorrect

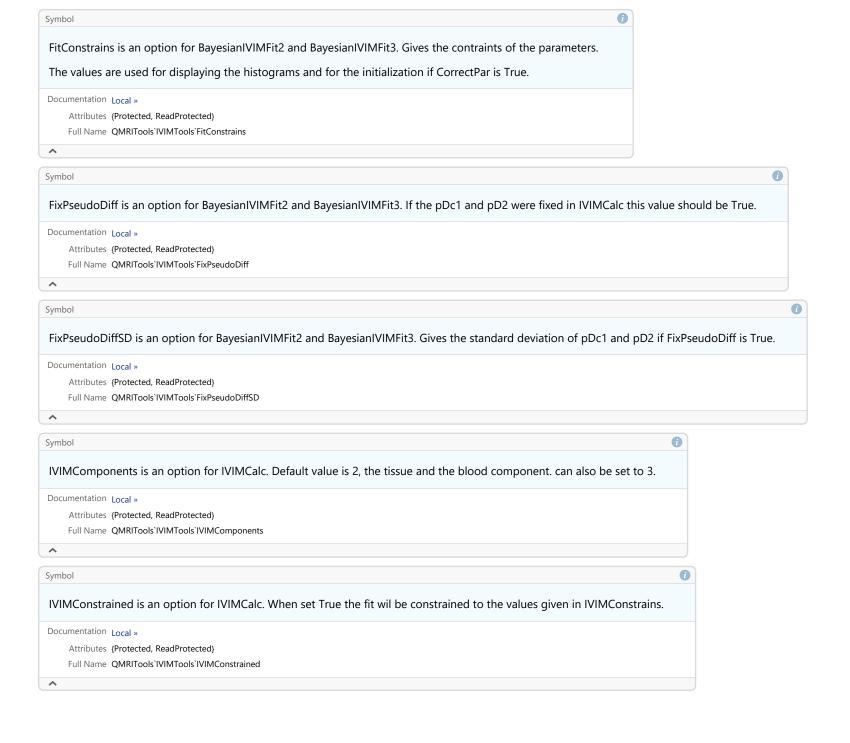
```
Symbol
HistogramPar[data, {constraints, Nbins}, style, color, range] plots histograms of IVIM solution.
HistogramPar[data, {constraints, Nbins, mu, conv}, components, color, range] plots histograms of IVIM solution.
data is {f1, dc, pdc1} or {f1, f2, dc, pdc1, pdc2}.
constraints are the ranges of the x-axes for the plots.
Nbins are the number of histogram bins.
style is the plot type, can be 1, 2, or 3.
color is the color of the histogram.
range are the ranges of the y-axes.
output is a row of histograms.
  Documentation Local »
Default Definitions SyntaxInformation[HistogramPar] = {ArgumentsPattern \rightarrow {_, _, _, _, _,}}
       Attributes {Protected, ReadProtected}
       Full Name QMRITools`IVIMTools`HistogramPar
\wedge
Symbol
IVIMCalc[data, binp, init] calculates the IVIM fit.
data should be 1D, 2D, 3D or 4D.
binp should be full bmatrix which can be calculated from the bvecs en bvals using Bmatrix with the bvalues in s/mm^2.
init should are the initialization parameters for 2 components this is {s0, f, D, Dp} for 3 componentes this is {s0, f1, f2, D, Dp1, Dp2}.
The fraction is defined between 0 and 1, the D, Dp, Dp1 and Dp2 is in mm^2/s.
output is {s0, f1, D, pD1} or {s0, f1, f2, D, pD1, pD2}.
  Documentation Local »
Default Definitions SyntaxInformation[IVIMCalc] = {ArgumentsPattern → {__ _, _, _, _, OptionsPattern[]}}
      Options > Method → Automatic ... (8 total)
       Attributes {Protected, ReadProtected}
       Full Name QMRITools`IVIMTools`IVIMCalc
```

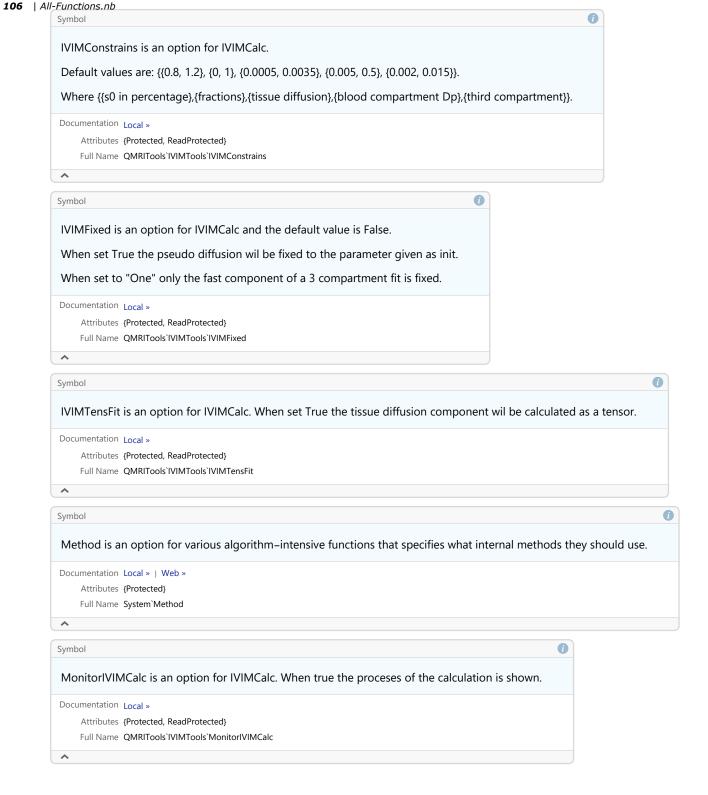
102 | All-Functions.nb 0 Symbol IVIMCorrectData[data, {s0, f, pdc}, bval] removes the ivim signal from the data. data is the original data. {s0, f, pdc} are the solution to a 2 compartment IVIM fit using IVIMCalc or BayesianIVIMFit2. bval are the bvalues. The fraction is defined between 0 and 1, the pdc is in mm²/s. output is the corrected data. Documentation Local » $Default\ Definitions\ \ SyntaxInformation[IVIMCorrectData] = \{ArgumentsPattern \rightarrow \{_, \{_, _, _\}, _, OptionsPattern[]\}\}$ Options {FilterMaps \rightarrow True, FilterType \rightarrow Median, FilterSize \rightarrow 1} Attributes {Protected, ReadProtected} Full Name QMRITools`IVIMTools`IVIMCorrectData ^ Symbol





Options





```
All-Functions.nb | 107
Symbol
OutputSamples is an option for BayesianIVIMFit2 and BayesianIVIMFit3. If set True the full marcov chain is given as an additionaln output.
Documentation Local »
    Attributes {Protected, ReadProtected}
    Full Name QMRITools`IVIMTools`OutputSamples
\wedge
Symbol
Parallelize[expr] evaluates expr using automatic parallelization.
Definitions
Parallelize[Parallel`Kernels`Private`args$___] := (Parallel`Protected`doAutolaunch[TrueQ[Parallel`Static`$enableLaunchFeedback]];
Parallelize[Parallel`Kernels`Private`args$])
Documentation Local » | Web »
      Options {DistributedContexts → $Context, Method → Automatic, ProgressReporting} → $ProgressReporting}
    Attributes {HoldFirst, Protected}
    Full Name System'Parallelize
\wedge
                                                                                                                                                                                  0
Symbol
UpdateStep is an option for BayesianIVIMFit2 and BayesianIVIMFit3. It determines how often the parameters are updated. Is optimized during the first 500 burn steps.
Documentation Local »
    Attributes {Protected, ReadProtected}
```

JcouplingTools

Full Name QMRITools`IVIMTools`UpdateStep

Functions

```
GetSpinSystem[name] get a spinsystem that can be used in SimHamiltonian. Current implementes systems are "glu", "lac", "gaba", "fatGly", "fatAll", "fatEnd", "fatDouble", "fatStart", and "fatMet".

Documentation Local »

Default Definitions SyntaxInformation[GetSpinSystem] = {ArgumentsPattern → {_, OptionsPattern[]}}

Options CenterFrequency → 0

Attributes {Protected, ReadProtected}}

Full Name QMRITools'JcouplingTools'GetSpinSystem
```

The output is a new spinsystem dout.

Attributes {Protected, ReadProtected}

Full Name QMRITools'JcouplingTools'SequenceSpaceEcho

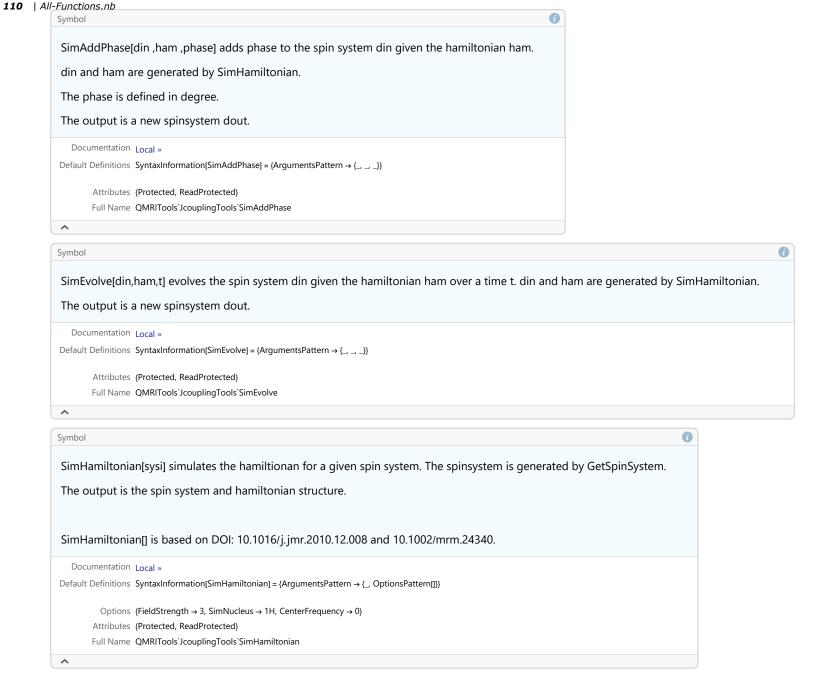
Default Definitions SyntaxInformation[SequenceSpaceEcho] = {ArgumentPattern → {__ _, _, _, _,}}

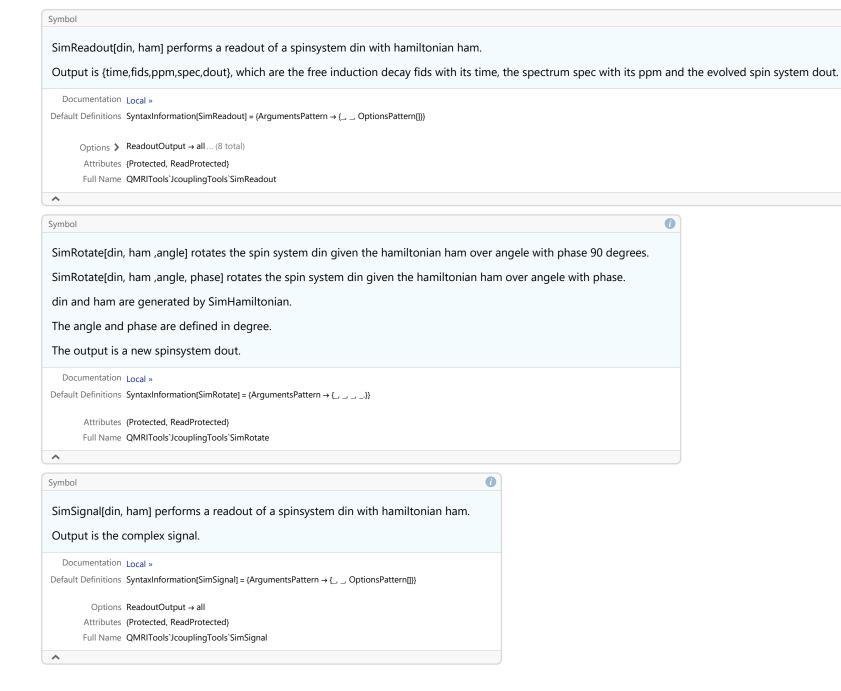
Documentation Local »

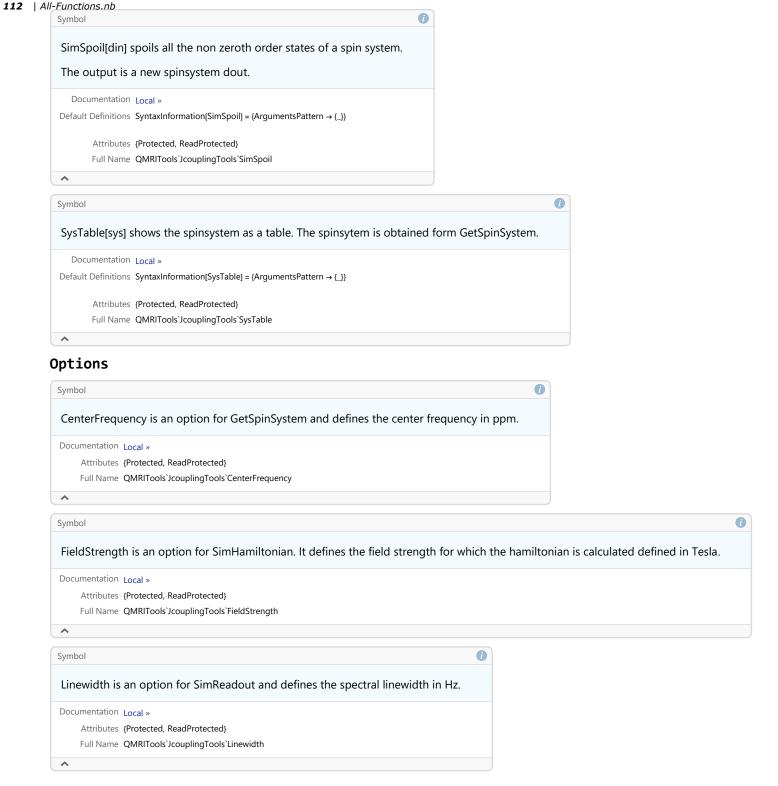
^

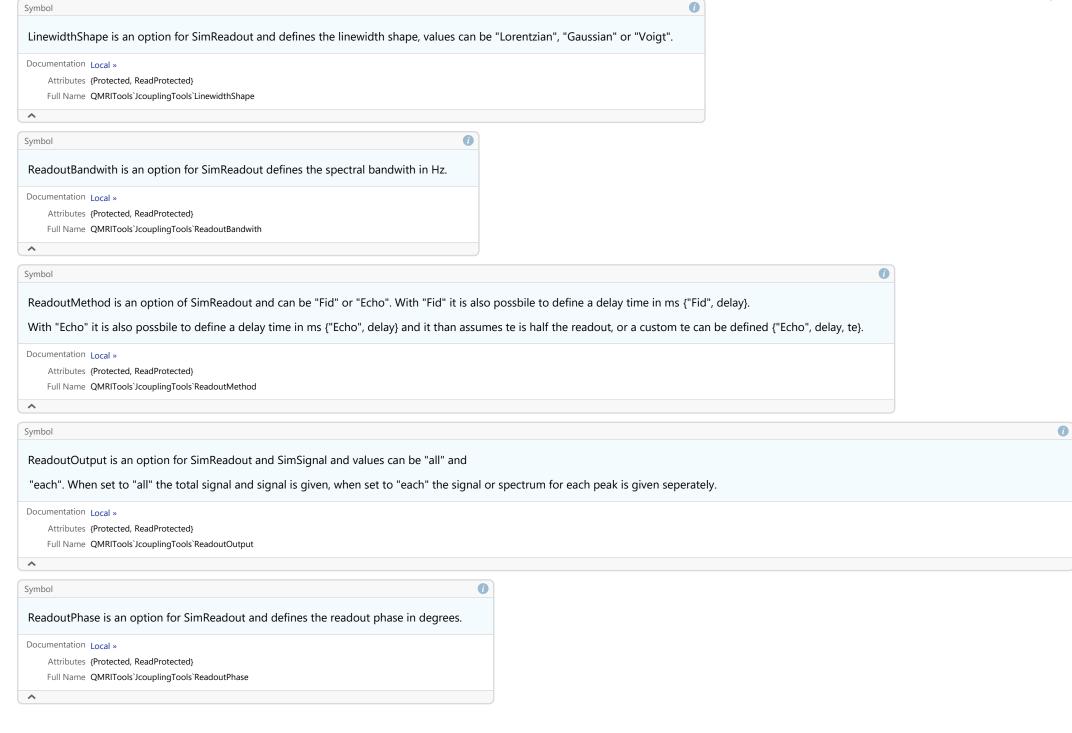
```
All-Functions.nb | 109
                                                                                                                                                                                                    0
Symbol
SequenceSpinEcho[din, ham, te] performs a spin echo experiment with echo time te of the spin system din given the hamiltonian ham with a 90 and 180 Degree pulse.
SequenceSpinEcho[din, ham, te, b1] performs a spin echo experiment with echo time te of the spin system din given the hamiltonian ham with a 90 and 180 Degree pulse and b1.
The te is defined in ms and the b1 of 100% is defined as 1.
The output is a new spinsystem dout.
  Documentation Local »
 Default \ Definitions \ \ SyntaxInformation[SequenceSpinEcho] = \{ArgumentsPattern \rightarrow \{\_,\_,\_.\}\} 
       Attributes (Protected, ReadProtected)
      Full Name QMRITools'JcouplingTools'SequenceSpinEcho
^
Symbol
SequenceSteam[din, ham, {te, tm}] performs a stimulated echo experiment with echo time te and mixing time tm of the spin system din given the hamiltonian ham with 3 90 Degree pulses.
The te and tm are defined in ms.
The output is a new spinsystem dout.
  Documentation Local »
Default Definitions SyntaxInformation[SequenceSteam] = {ArgumentsPattern → {_, _, {_, _}}}}
       Attributes {Protected, ReadProtected}
      Full Name QMRITools'JcouplingTools'SequenceSteam
^
Symbol
SequenceTSE[din ,ham, {te, necho}, {ex, ref}] performs a multi echo spin echo experiment with
echo time te with necho echos of the spin system din given the hamiltonian ham using ex Degree exitation and ref Degree refocus pulses.
SequenceTSE[din ,ham, {te, necho}, {ex, ref}, b1] performs a multi echo spin echo experiment with echo
time te with necho echos of the spin system din given the hamiltonian ham using ex Degree exitation and ref Degree refocus pulses and b1.
The te is defined in ms, the ex and ref are defined in degree and b1 of 100% is defined as 1.
The output is a new spinsystem dout.
  Documentation Local »
Default\ Definitions\ \ SyntaxInformation[SequenceTSE] = \{ArgumentsPattern \rightarrow \{\_, \_, \{\_, \_, \_\}, \{\_, \_\}\}\}
       Attributes {Protected, ReadProtected}
       Full Name QMRITools'JcouplingTools'SequenceTSE
```

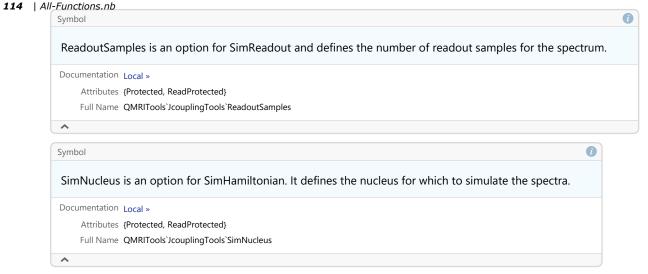
^



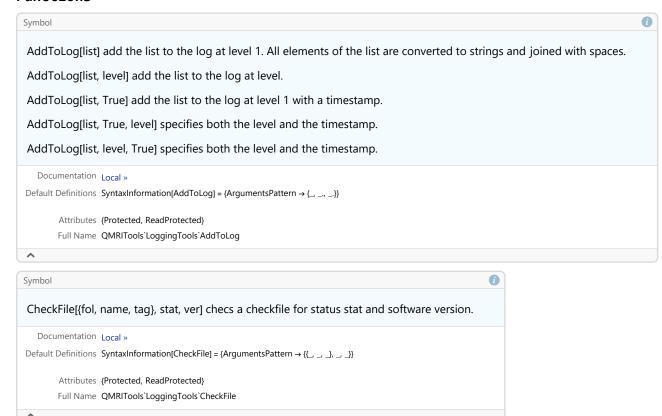


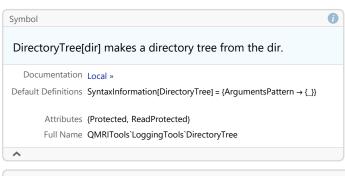


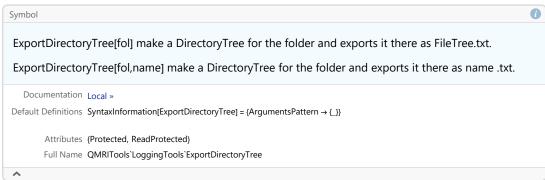


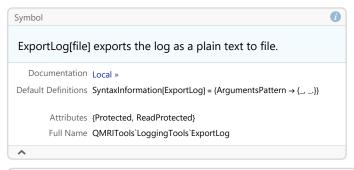


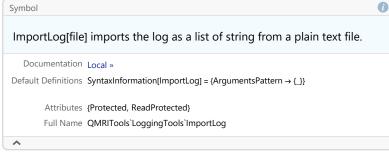
LoggingTools

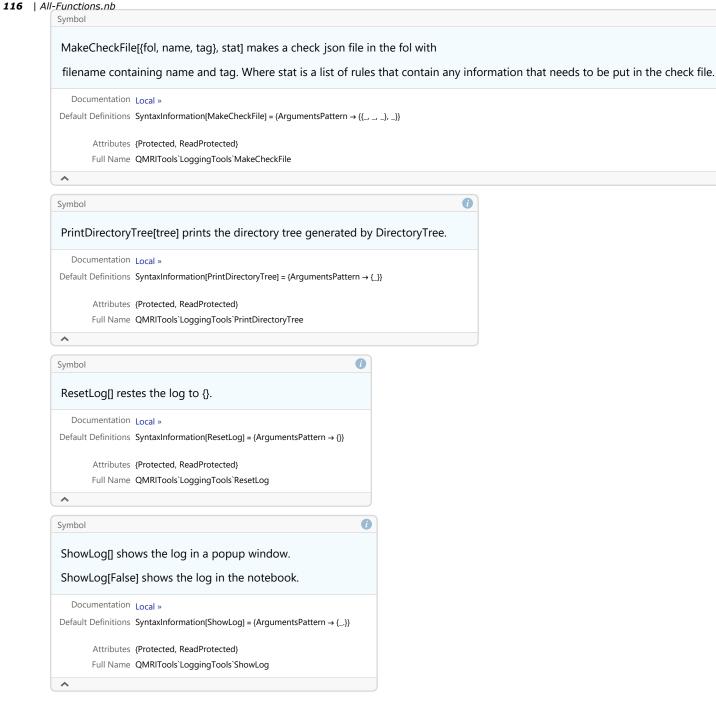






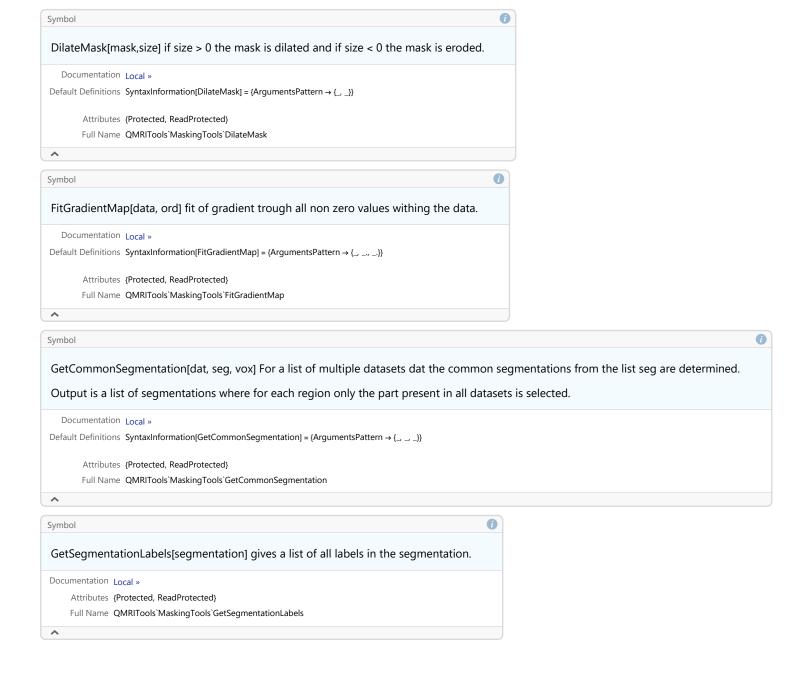


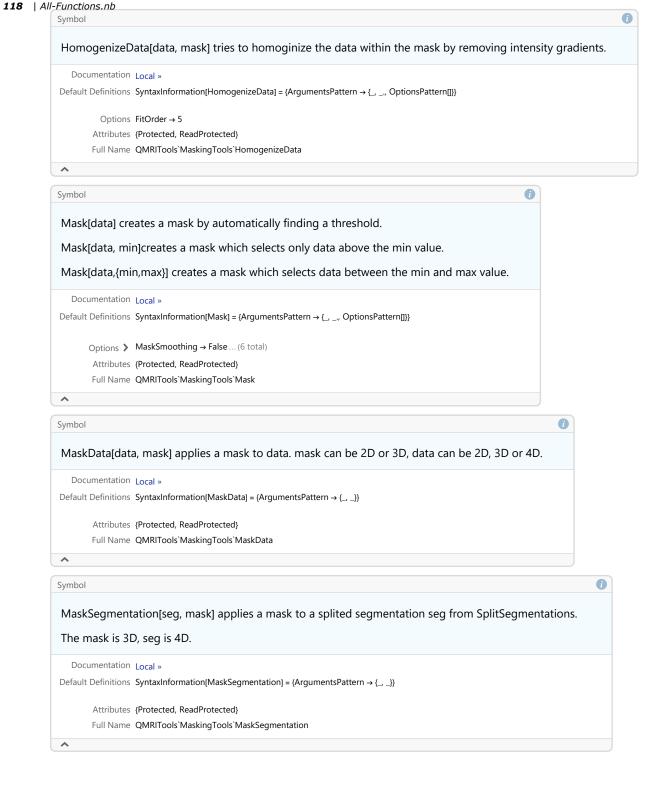




Options

MaskingTools

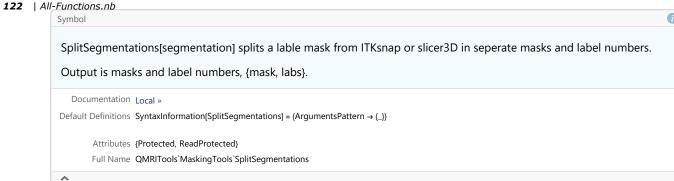


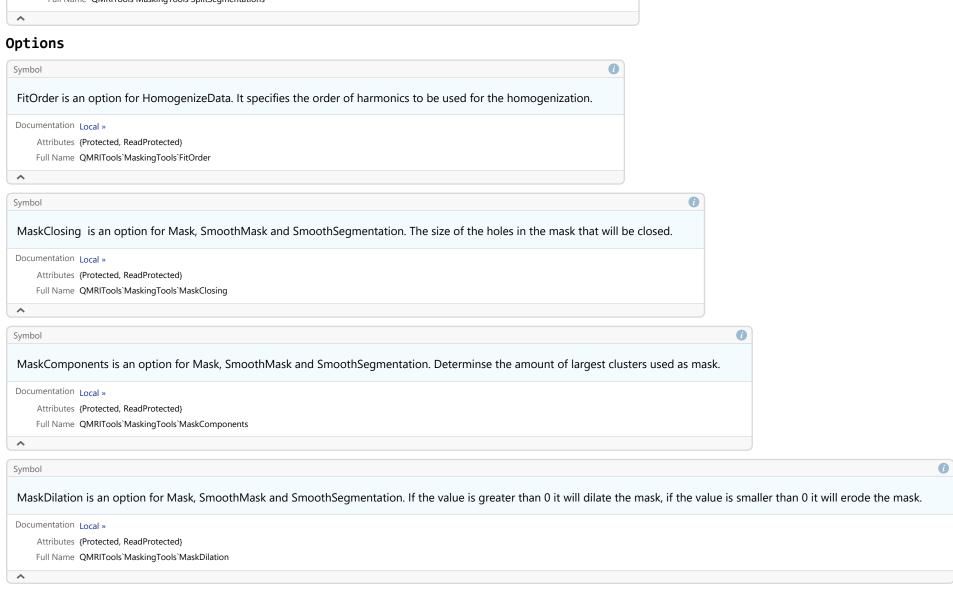


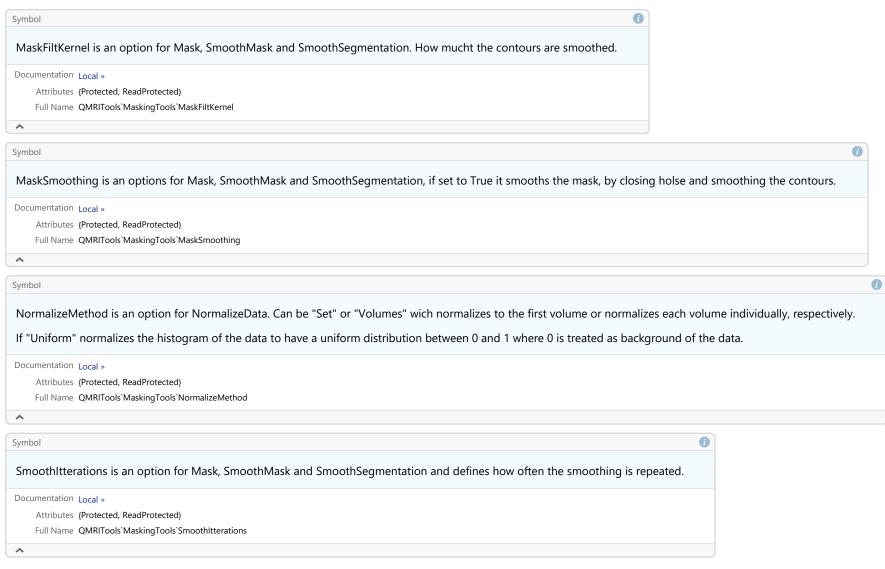
```
Symbol
MergeSegmentations[masks, labels] generates an ITKsnap or slices3D compatible segmentation from individual masks and label numbers.
Output is a labled segmentation.
MergeSegmentations[masks] does the same but automatically numbers the segmentations.
  Documentation Local »
Default Definitions SyntaxInformation[MergeSegmentations] = {ArgumentsPattern \rightarrow {_, _.}}
       Attributes {Protected, ReadProtected}
       Full Name QMRITools`MaskingTools`MergeSegmentations
^
Symbol
NormalizeData[data] normalizes the data to the mean signal of the data. For 4D data it normalizes to the first volume of the 4th dimension.
NormalizeData[data, mask] normalizes the data based on the mean signal only within the mask.
  Documentation Local »
Default Definitions SyntaxInformation[NormalizeData] = {ArgumentsPattern → {_, __, OptionsPattern[]}}
         Options NormalizeMethod → Set
       Attributes {Protected, ReadProtected}
       Full Name QMRITools`MaskingTools`NormalizeData
^
                                                                                                 0
Symbol
NormalizeMeanData[data] calculates the mean normalized data from a 4D dataset.
  Documentation Local »
Default Definitions SyntaxInformation[NormalizeMeanData] = {ArgumentsPattern → {_, _, _, OptionsPattern[]}}
         Options NormalizeMethod → Set
       Attributes {Protected, ReadProtected}
       Full Name QMRITools`MaskingTools`NormalizeMeanData
\wedge
Symbol
RemoveMaskOverlaps[mask] removes the overlaps between multiple masks. Mask is a 4D dataset with {z, masks, x, y}.
  Documentation Local »
Default\ Definitions\ \ SyntaxInformation[RemoveMaskOverlaps] = \{ArgumentsPattern \rightarrow \{\_\}\}
       Attributes {Protected, ReadProtected}
       Full Name QMRITools`MaskingTools`RemoveMaskOverlaps
\wedge
```

^

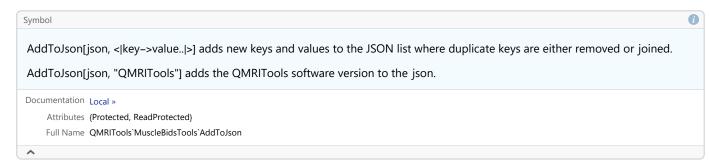
```
Symbol
SelectMaskComponents[mask] selects the largest connected component in the mask.
SelectMaskComponents[mask,n] selects the n largest connected components in the mask.
Default\ Definitions\ \ SyntaxInformation[SelectMaskComponents] = \{ArgumentsPattern \rightarrow \{\_,\_\}\}
       Attributes {Protected, ReadProtected}
       Full Name QMRITools`MaskingTools`SelectMaskComponents
\wedge
                                                                                                                0
Symbol
SelectSegmentations[seg, labs] selects only the segmentaions from seg with label number labs.
  Documentation Local »
Default\ Definitions\ \ SyntaxInformation[SelectSegmentations] = \{ArgumentsPattern \rightarrow \{\_, \_\}\}
       Attributes {Protected, ReadProtected}
       Full Name QMRITools`MaskingTools`SelectSegmentations
\wedge
                                                                                          0
Symbol
SmoothMask[mask] generates one clean masked volume form a noisy mask.
  Documentation Local »
Default Definitions SyntaxInformation[SmoothMask] = {ArgumentsPattern → {_, OptionsPattern[]}}
      Options ➤ MaskComponents → 1... (5 total)
       Attributes {Protected, ReadProtected}
       Full Name QMRITools`MaskingTools`SmoothMask
^
Symbol
SmoothSegmentation[segmentation] smooths segmentations and removes the overlaps between multiple segmentations.
SmoothSegmentation[segmentation, labs] only smooths the selected label number labs.
  Documentation Local »
Default Definitions SyntaxInformation[SmoothSegmentation] = {ArgumentsPattern → {___, OptionsPattern[]}}
      Options ➤ MaskComponents → 1 ... (5 total)
       Attributes {Protected, ReadProtected}
       Full Name QMRITools`MaskingTools`SmoothSegmentation
^
```

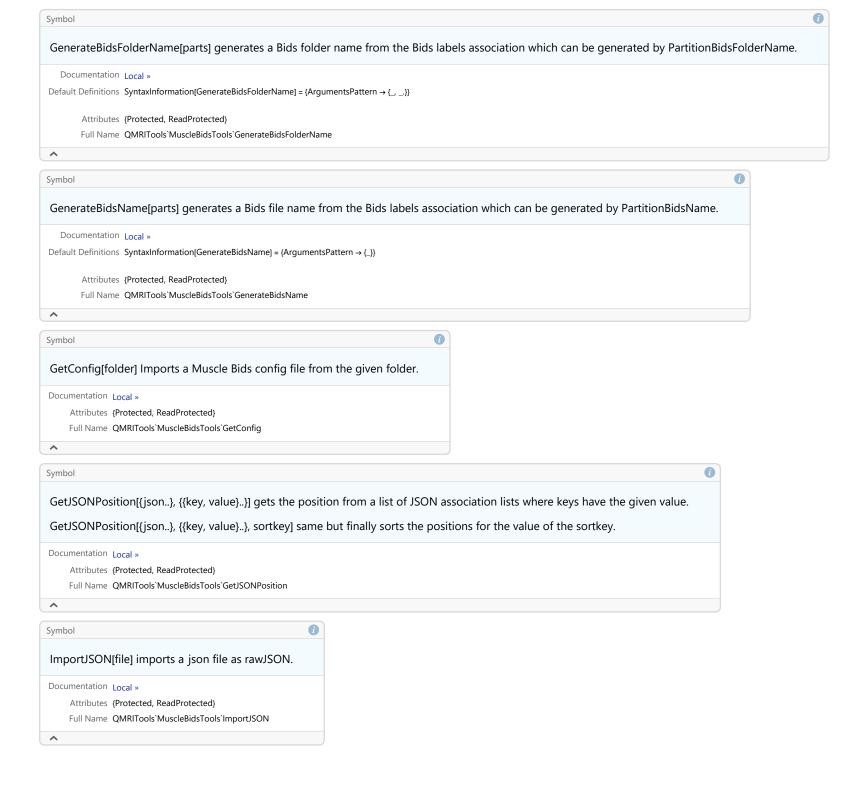






MuscleBidsTools



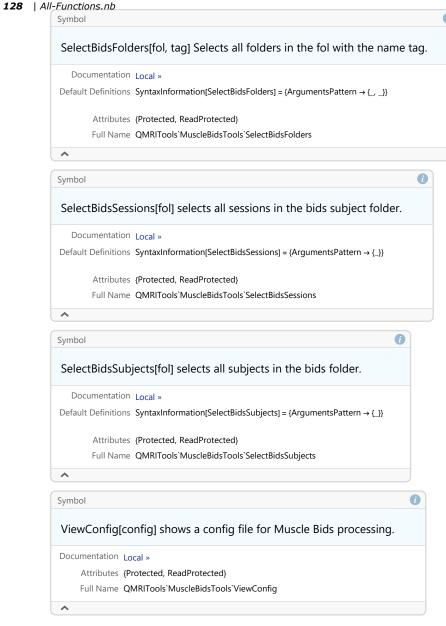


PartitionBidsName[name] converts a Bids name to the Bids labels as an association, i.e. {"sub","ses","stk","rep","type","suf"}.

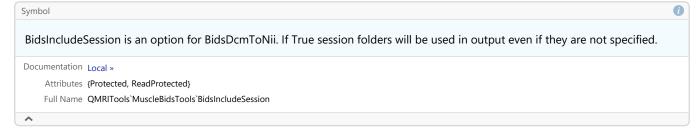
Documentation Local »

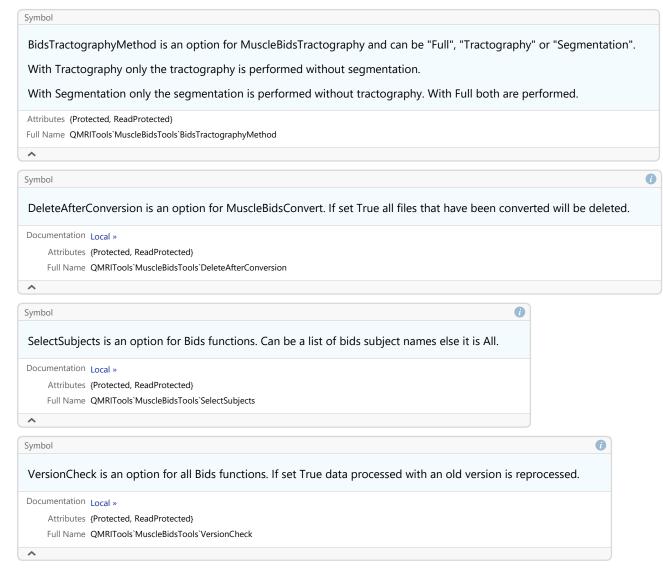
Default Definitions SyntaxInformation[PartitionBidsName] = {ArgumentsPattern → {_}}}

Attributes {Protected, ReadProtected}
Full Name QMRITools`MuscleBidsTools`PartitionBidsName

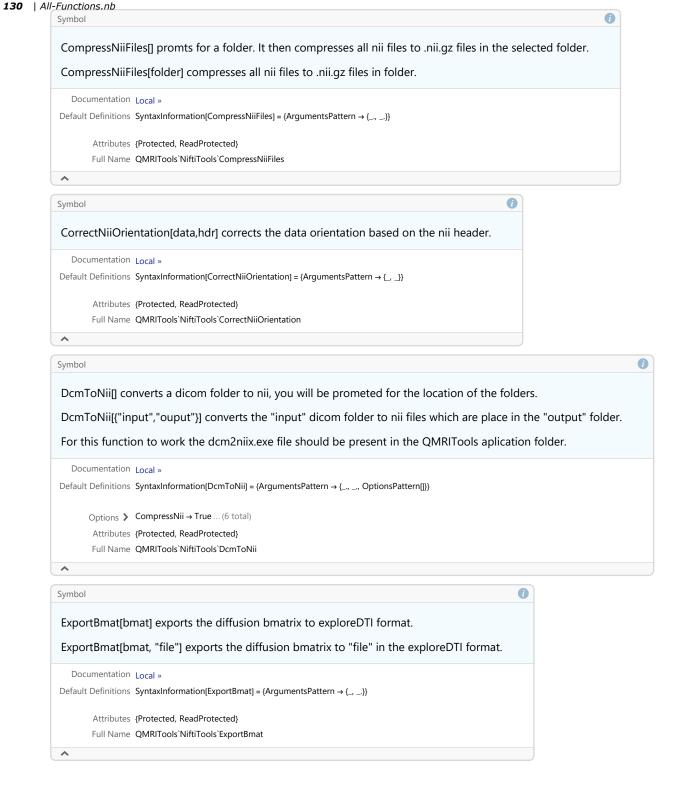


Options

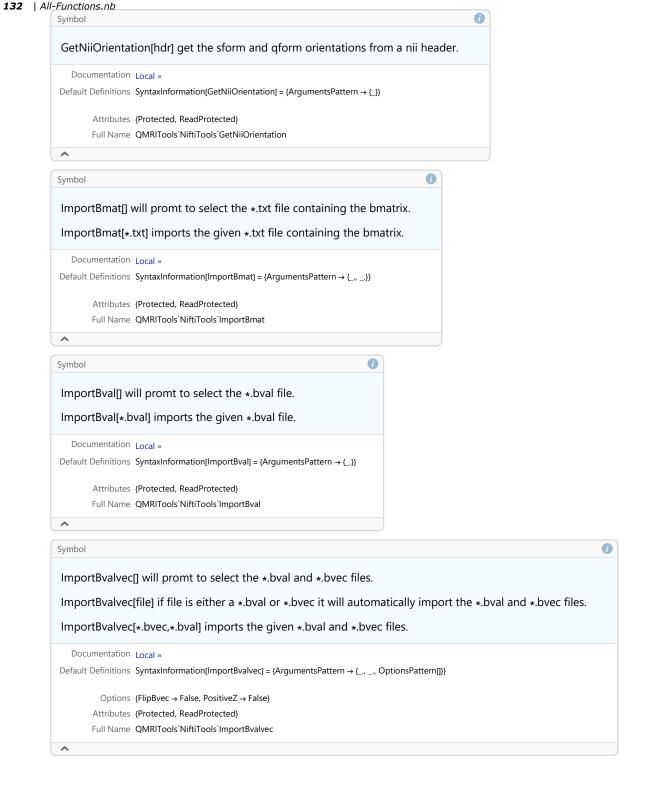




NiftiTools



```
0
Symbol
ExportBval[bvals] exports the diffusion bvalues to exploreDTI format.
ExportBval[bvals, "file"] exports the diffusion bvalues to "file" in the exploreDTI format.
  Documentation Local »
Default Definitions SyntaxInformation[ExportBval] = {ArgumentsPattern \rightarrow {_, __}}
        Attributes {Protected, ReadProtected}
       Full Name QMRITools`NiftiTools`ExportBval
\wedge
Symbol
ExportBvec[grad] exports the diffusion gradients to exploreDTI format.
ExportBvec[grad, "file"] exports the diffusion gradients to "file" in the exploreDTI format.
  Documentation Local »
Default Definitions SyntaxInformation[ExportBvec] = {ArgumentsPattern → {_, _,, OptionsPattern[]}}
          Options {FlipBvec → False, PositiveZ → False}
        Attributes {Protected, ReadProtected}
        Full Name QMRITools`NiftiTools`ExportBvec
^
Symbol
ExportNii[data, vox] exports the nii file and will promt for a file name.
ExportNii[data, vox, "file"] exports the nii file to the location "file".
  Documentation Local »
Default Definitions SyntaxInformation[ExportNii] = {ArgumentsPattern \rightarrow {__ ___, OptionsPattern[]}}
       Options ➤ NiiDataType → Automatic ... (5 total)
        Attributes {Protected, ReadProtected}
        Full Name QMRITools`NiftiTools`ExportNii
^
Symbol
ExtractNiiFiles[] promts for a folder. It then extracts all nii.gz files to .nii files in the selected folder.
ExtractNiiFiles[folder] extracts all nii.gz files to .nii files in folder.
  Documentation Local »
\label{eq:definitions} Default\ Definitions\ \ SyntaxInformation[ExtractNiiFiles] = \{ArgumentsPattern \rightarrow \{\_,\_\}\}
        Attributes {Protected, ReadProtected}
        Full Name QMRITools`NiftiTools`ExtractNiiFiles
\wedge
```



ImportNii[] promts to select the nii file to import.

ImportNii["file"] imports the nii file.

The default output is {data, vox}, however using NiiMethod various outputs can be given.

The Nii import is also suported using the native Import function from Mathematica.

Documentation Local »

Default Definitions SyntaxInformation[ImportNii] = {ArgumentsPattern → {_, OptionsPattern[]}}

Options {NiiMethod → default, NiiScaling → False}

Attributes {Protected, ReadProtected}

Full Name QMRITools`NiftiTools`ImportNii

^

All-Functions.nb | 133

```
MakeNiiOrentationS[off, vox] maxes the srow values for nii header assuming not rot and Q.

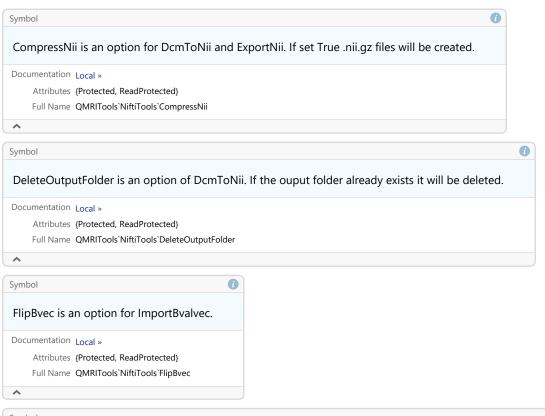
MakeNiiOrentationS[off, vox, rot] maxes the srow values for nii header using rotation rot.

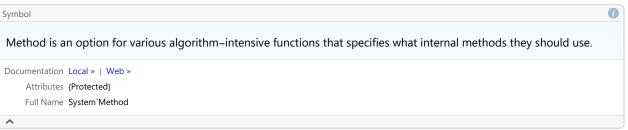
MakeNiiOrentationS[off, vox, rot, Q] maxes the srow values for nii header using rotation rot and skew Q.

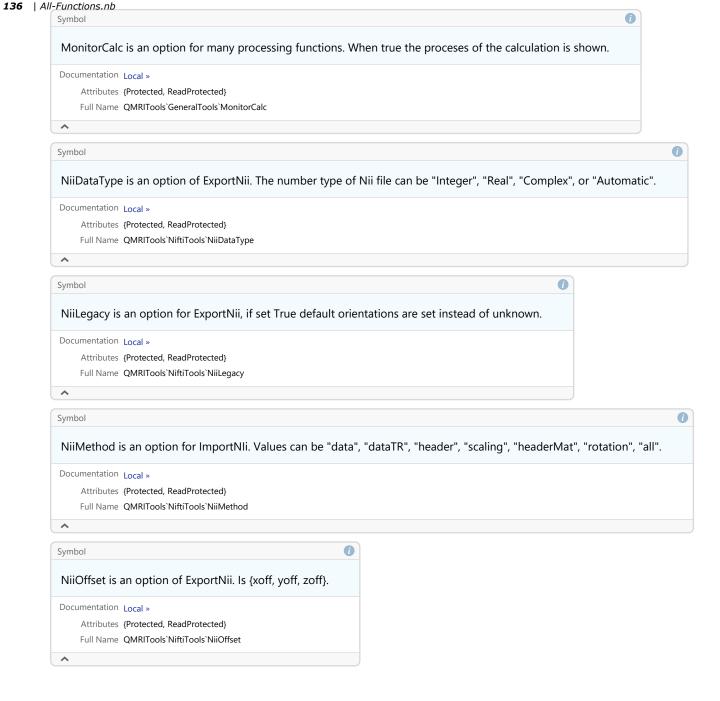
Documentation Local »

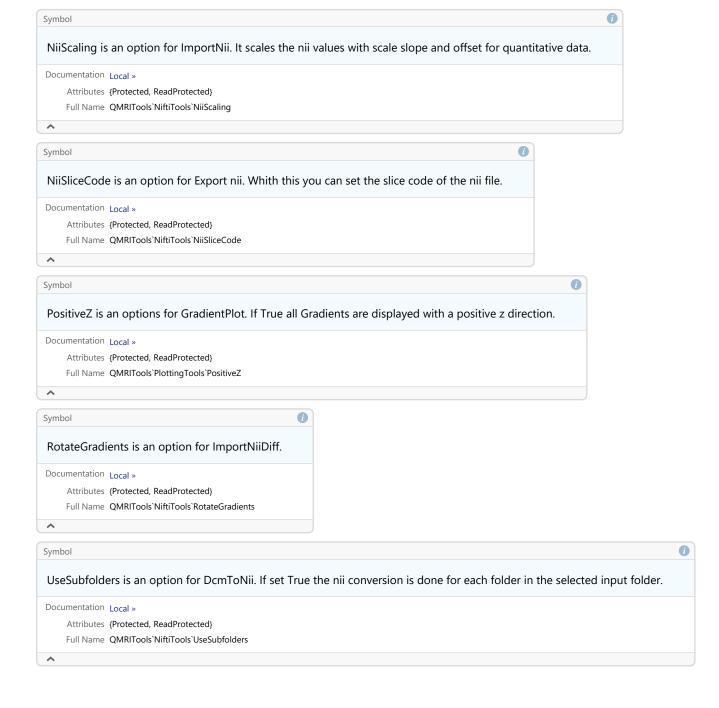
Attributes (Protected, ReadProtected)
Full Name QMRITools'NiftiTools'MakeNiiOrentationS
```

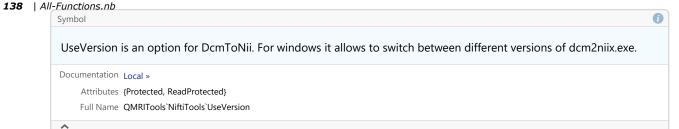
Options

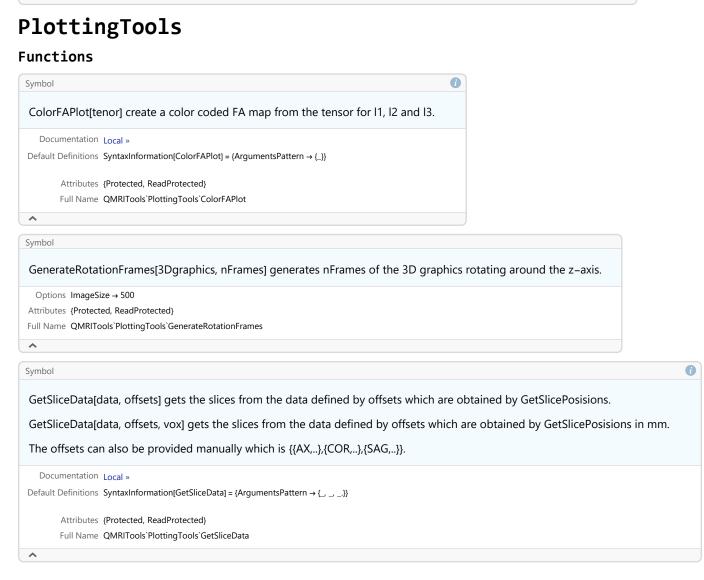




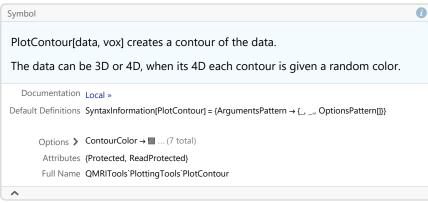








```
Symbol
GetSlicePositions[data] finds the position of slices with the maximal signal in voxel index.
GetSlicePositions[data, vox] find the position of slices with the maximal signal in mm.
   Documentation Local »
Default Definitions SyntaxInformation[GetSlicePositions] = {ArgumentsPattern → {___, OptionsPattern[]}}
          Options \{MakeCheckPlot \rightarrow False, DropSlices \rightarrow \{1, 1, 1\}, PeakNumber \rightarrow \{1, 1, 2\}\}
        Attributes {Protected, ReadProtected}
        Full Name QMRITools`PlottingTools`GetSlicePositions
^
                                                                                                                                    0
Symbol
GradientPlot[bvec, bval] plots the given bvec with position of the gradients scaled according to the bval.
   Documentation Local »
Default Definitions SyntaxInformation[GradientPlot] = {ArgumentsPattern \rightarrow {_, _,, OptionsPattern[]}}
       Options ➤ PlotSpace → bspace ... (4 total)
        Attributes {Protected, ReadProtected}
        Full Name QMRITools`PlottingTools`GradientPlot
\wedge
Symbol
Link3DGraphic[3Dgraphics] creates a 3D graphic with a linked view options between all 3D graphics that have been linked.
Attributes {Protected, ReadProtected}
Full Name QMRITools`PlottingTools`Link3DGraphic
^
                                                                                              0
Symbol
ListSpherePlot[points] plots 3D points as spheres.
   Documentation Local »
Default Definitions SyntaxInformation[ListSpherePlot] = {ArgumentsPattern → {_, OptionsPattern[]}}
          Options {SphereSize \rightarrow 2, SphereColor \rightarrow Automatic}
        Attributes {Protected, ReadProtected}
        Full Name QMRITools`PlottingTools`ListSpherePlot
^
```



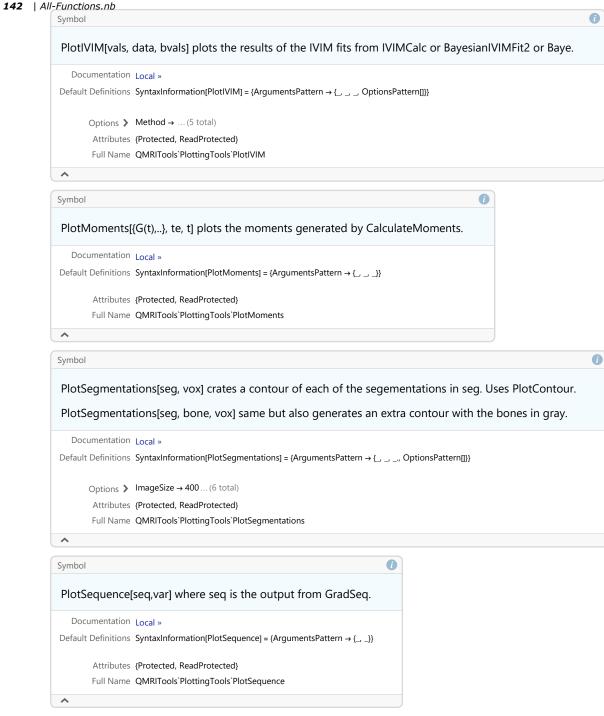
PlotCorrection[w] plots deformation vectors w {w1,w2...} generated by Registration2D and Registration3D for multiple datasets or registration steps.

Documentation Local »

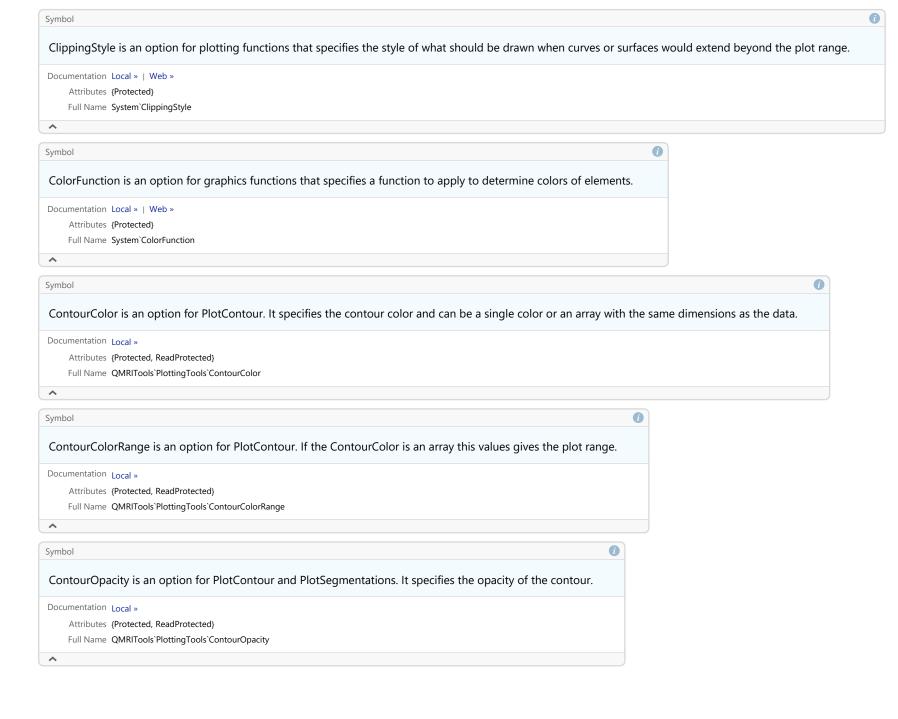
Default Definitions SyntaxInformation[PlotCorrection] = {ArgumentsPattern → {__}}}

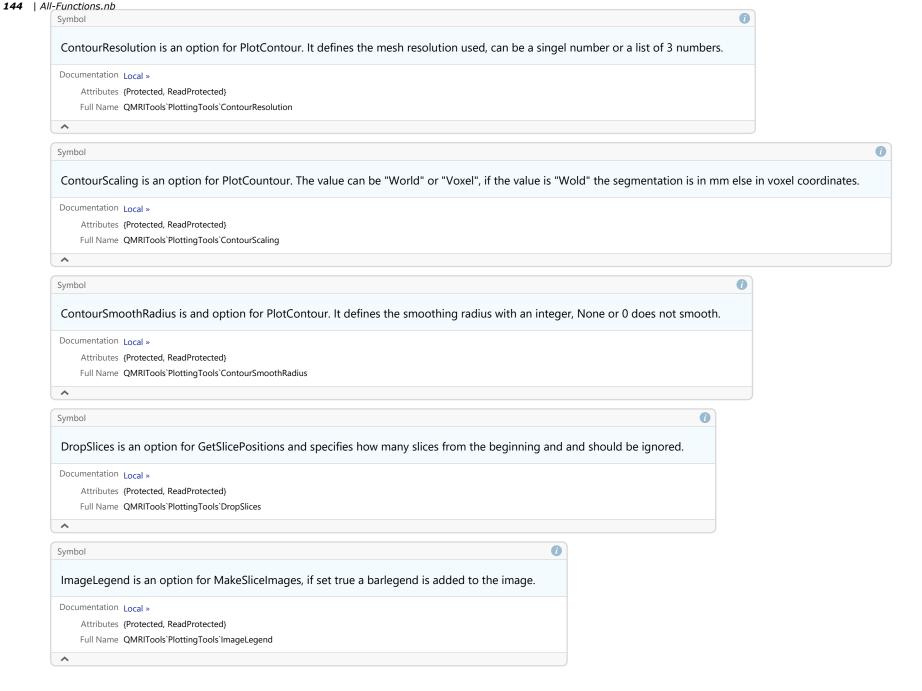
Attributes {Protected, ReadProtected}
Full Name QMRITools`PlottingTools`PlotCorrection

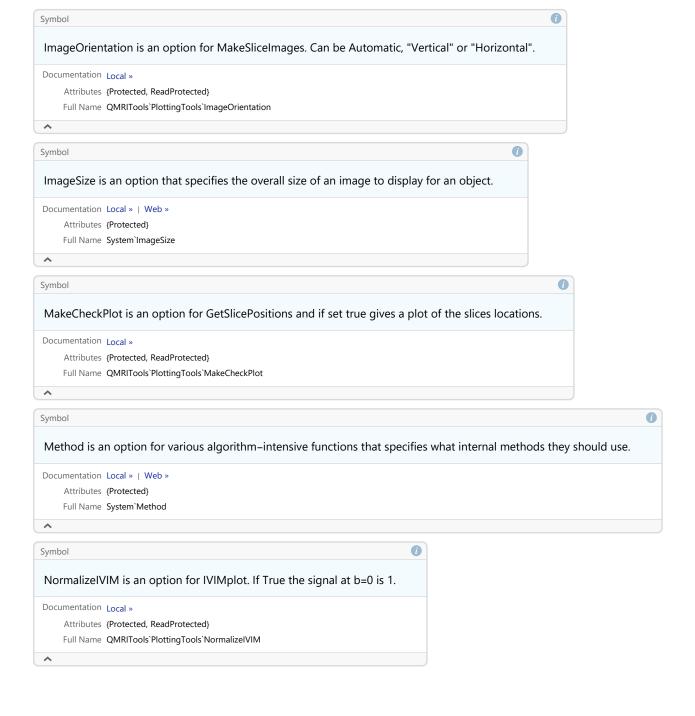
```
Symbol
PlotData[data] plots the data.
PlotData[data, vox] plots the data and for 3D and 4D data assumes the voxelsize vox (z,x,y).
PlotData[data1, data2] plots data1 and data2.
PlotData[data1, data2, vox] plots data1 and data2 and for 3D and 4D data assumes the voxelsize vox (z,x,y).
  Documentation Local »
Default Definitions SyntaxInformation[PlotData] = {ArgumentsPattern → {____, ___, OptionsPattern[]}}
         Options {PlotRange → Auto, ColorFunction → BlackToWhite}
       Attributes {Protected, ReadProtected}
       Full Name QMRITools`PlottingTools`PlotData
^
Symbol
PlotData3D[data,vox] is a 3D dataviewer, data is the 3D dataset and voxsize the size of the voxels in mm (z,x,y).
  Documentation Local »
Default Definitions SyntaxInformation[PlotData3D] = {ArgumentsPattern \rightarrow \{\_, \_.\}}
       Attributes {Protected, ReadProtected}
       Full Name QMRITools`PlottingTools`PlotData3D
^
                                                                                                                                                                              0
Symbol
PlotDefGrid[data, phasemap, shiftpar] plots the dataset on the background with on top the non deformed and the deformed grid, or arrows or lines.
Documentation Local »
    Attributes {Protected, ReadProtected}
    Full Name QMRITools`PlottingTools`PlotDefGrid
^
                                                                           0
Symbol
PlotDuty[{grad, bval, ord}, mode] plot the gradient dutycycle.
  Documentation Local »
Default Definitions SyntaxInformation[PlotDuty] = {ArgumentsPattern \rightarrow {\_, \_, \_}}
       Attributes {Protected, ReadProtected}
       Full Name QMRITools`PlottingTools`PlotDuty
```

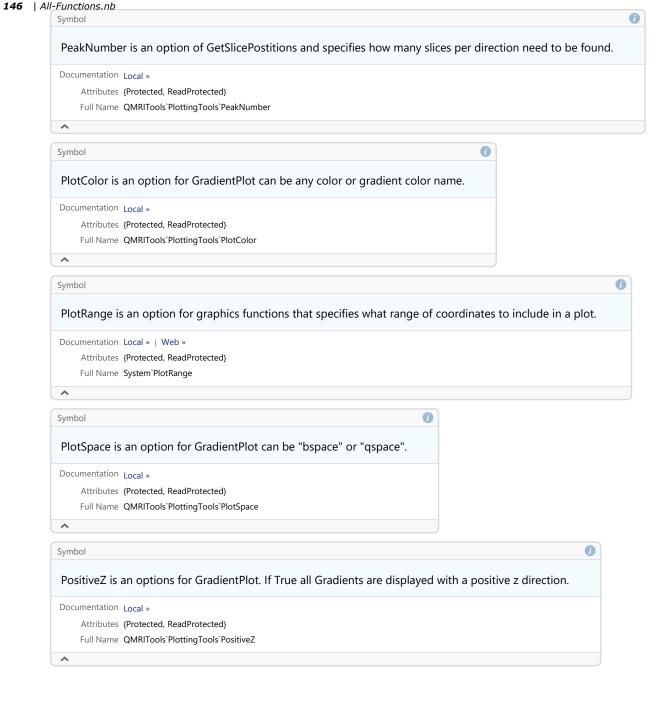


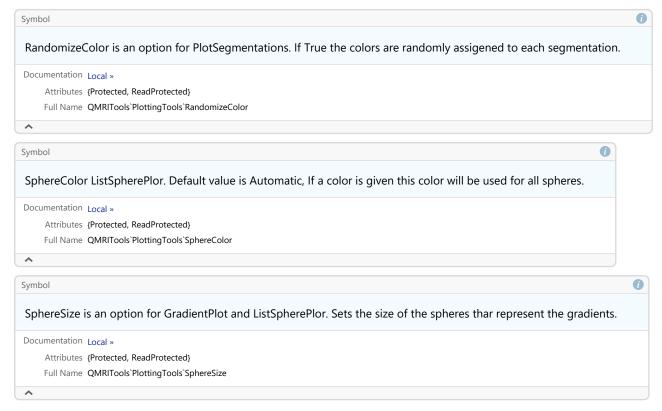
Options





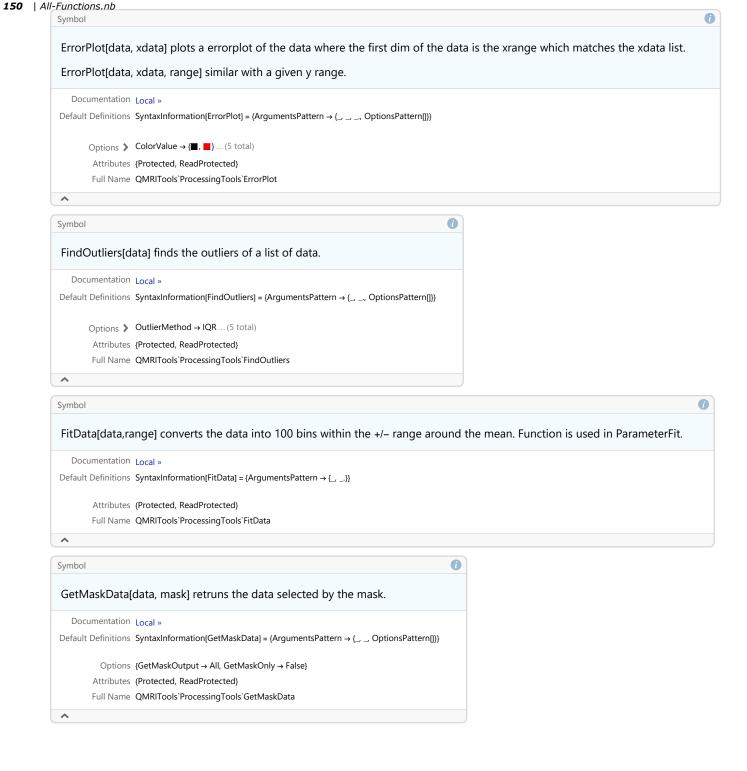


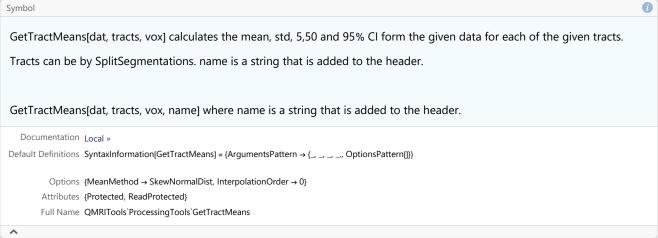




ProcessingTools

Functions





Hist[data, range] plots a probability density histogram of the data from xmin to xmax with a fitted (skew)normal distribution. Uses ParameterFit.

Hist[data, range, label] plots a probability density histogram of the data from xmin to xmax with a fitted (skew)normal distribution and label as x-axis label.

Hist[{data1..., data2...}, {range1, range2,...}] plots a probability density histogram of the data from xmin to xmax with a fitted (skew)normal distribution. Uses ParameterFit.

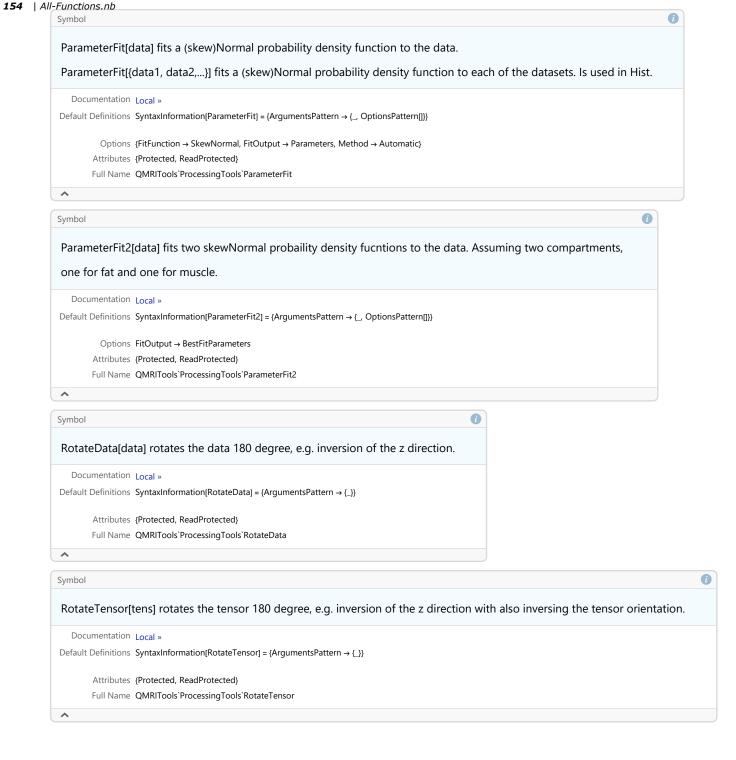
Hist[{data1, data2...}, {range1, range2,...}, {label1, label2,...}] plots a probability density histogram of the data from xmin to xmax with a fitted (skew)normal distribution and label as x-axis label.

Documentation Local >>

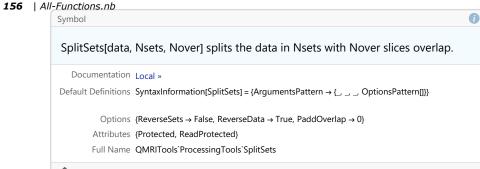
Default Definitions SyntaxInformation[Hist] = {ArgumentsPatterm → {..., OptionsPattern[]}}

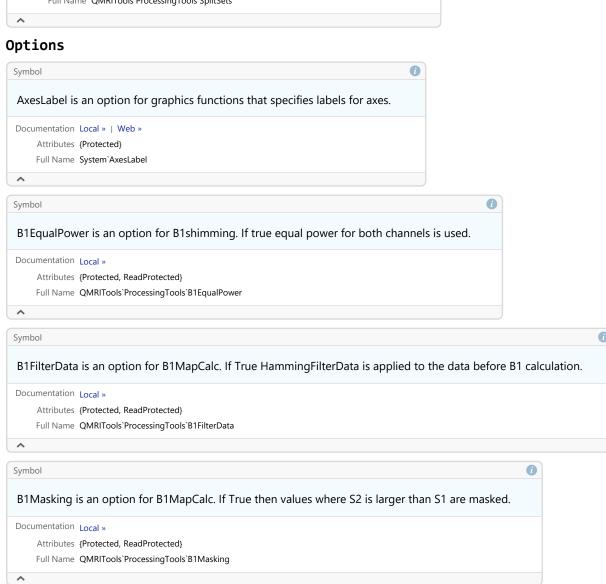
Options >> ColorValue → {(||...||, ||...||, ||...||, ||...||, ||...||, ||...||, ||...||, ||...||, ||...||, ||...||, ||...||, ||...||, ||...||, ||...||, ||...||, ||...||, ||...||, ||...||, ||...||, ||...||, ||...||, ||...||, ||...||, ||...||, ||...||, ||...||, ||...||, ||...||, ||...||, ||...||, ||...||, ||...||, ||...||, ||...||, ||...||, ||...||, ||...||, ||...||, ||...||, ||...||, ||...||, ||...||, ||...||, ||...||, ||...||, ||...||, ||...||, ||...||, ||...||, ||...||, ||...||, ||...||, ||...||, ||...||, ||...||, ||...||, ||...||, ||...||, ||...||, ||...||, ||...||, ||...||, ||...||, ||...||, ||...||, ||...||, ||...||, ||...||, ||...||, ||...||, ||...||, ||...||, ||...||, ||...||, ||...||, ||...||, ||...||, ||...||, ||...||, ||...||, ||...||, ||...||, ||...||, ||...||, ||...||, ||...||, ||...||, ||...||, ||...||, ||...||, ||...||, ||...||, ||...||, ||...||, ||...||, ||...||, ||...||, ||...||, ||...||, ||...||, ||...||, ||...||, ||...||, ||...||, ||...||, ||...||, ||...||, ||...||, ||...||, ||...||, ||...||, ||...||, ||...||, ||...||, ||...||, ||...||, ||...||, ||...||, ||...||, ||...||, ||...||, ||...||, ||...||, ||...||, ||...||, ||...||, ||...||, ||...||, ||...||, ||...||, ||...||, ||...||, ||...||, ||...||, ||...||,

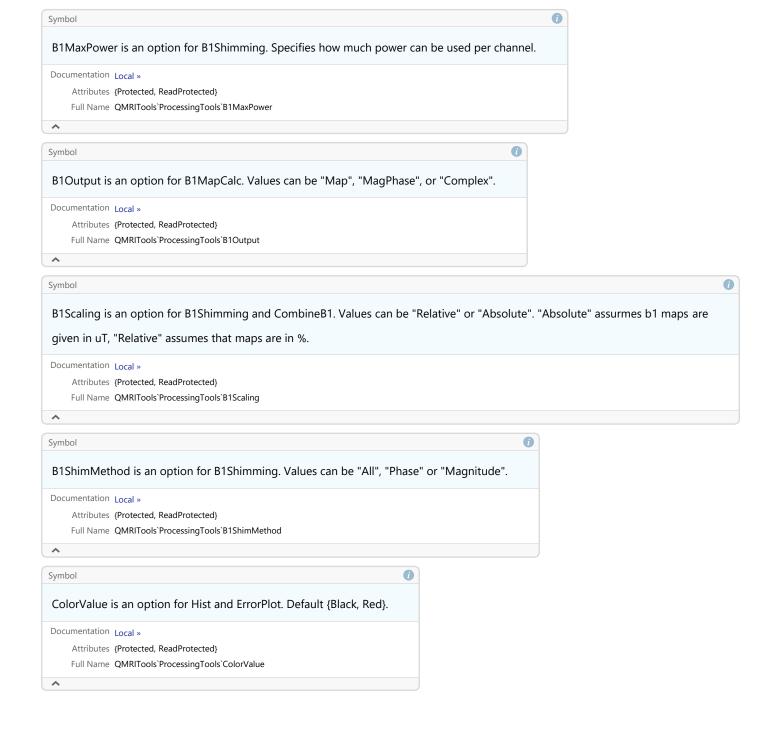
All-Functions.nb | 153

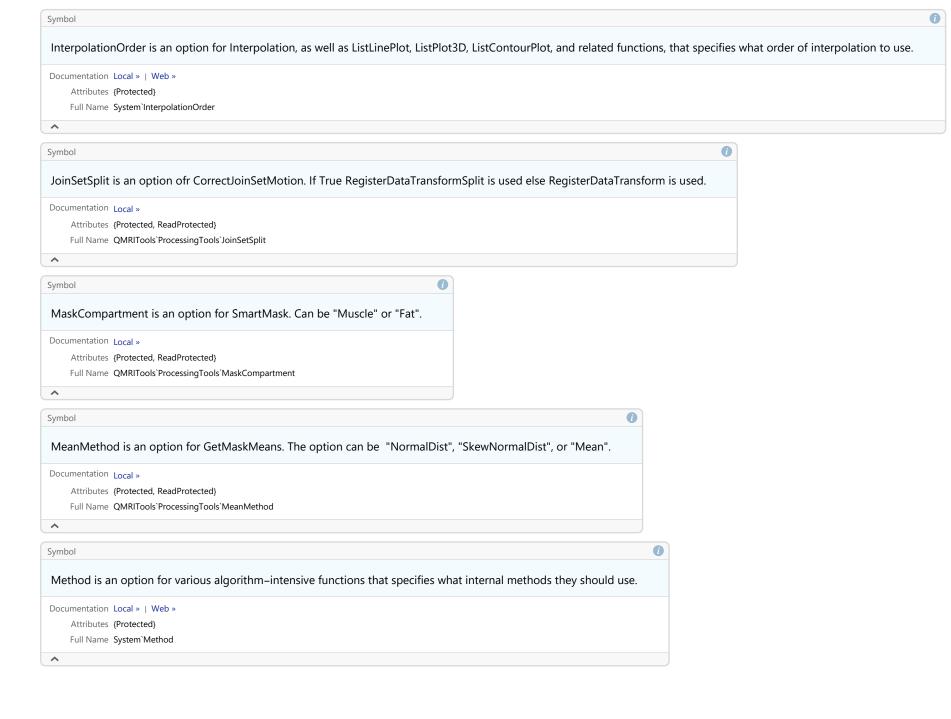


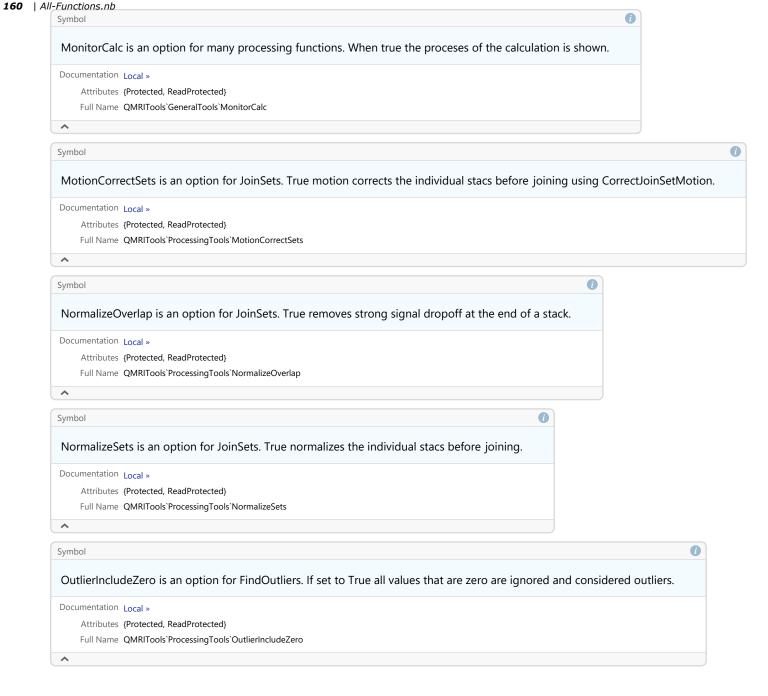
```
Symbol
SmartMask[input] crates a smart mask of input, which is either the tensor or the tensor parameters calculated using ParameterCalc.
SmartMask[input, mask] crates a smart mask of input and used the mask as a prior selection of the input.
  Documentation Local »
Default Definitions SyntaxInformation[SmartMask] = {ArgumentsPattern → {_, _,, OptionsPattern[]}}
      Options > Strictness → 0.5 ... (4 total)
       Attributes {Protected, ReadProtected}
       Full Name QMRITools`ProcessingTools`SmartMask
^
                                                                                                                                        0
Symbol
SNRCalc[data, sigma] calculates the ANR of the data using a noise sigma map for example generated by PCADeNoise.
  Documentation Local »
Default Definitions SyntaxInformation[SNRCalc] = \{ArgumentsPattern \rightarrow \{\_, \_\}\}\
       Attributes {Protected, ReadProtected}
       Full Name QMRITools`ProcessingTools`SNRCalc
^
                                                                                                                                                                   0
Symbol
SNRMapCalc[data1,noisemap] calcualtes the signal to noise ratio of the data using MN[data]/(1/sqrt[pi/2] sigma),
where sigma is the local mean of the noise map assuming it is a rician distribution.
SNRMapCalc[{data1,data2}] calcualtes the signal to noise ratio from two identical images using
MN[data1,data2] / (.5 SQRT[2] STDV[data2-data1]).
SNRMapCalc[{data1, .. dataN}] calcualtes the signal to noise ratio of the data using MN/sigma where the mean signal MN is the average voxel
value over all dynamics N and the sigma is the standard deviation over all dynamics N.
  Documentation Local »
Default\ Definitions\ \ SyntaxInformation[SNRMapCalc] = \{ArgumentsPattern \rightarrow \{\_,\_,\_,OptionsPattern[]\}\}
         Options {OutputSNR \rightarrow SNR, SmoothSNR \rightarrow 2}
       Attributes {Protected, ReadProtected}
       Full Name QMRITools`ProcessingTools`SNRMapCalc
\wedge
```

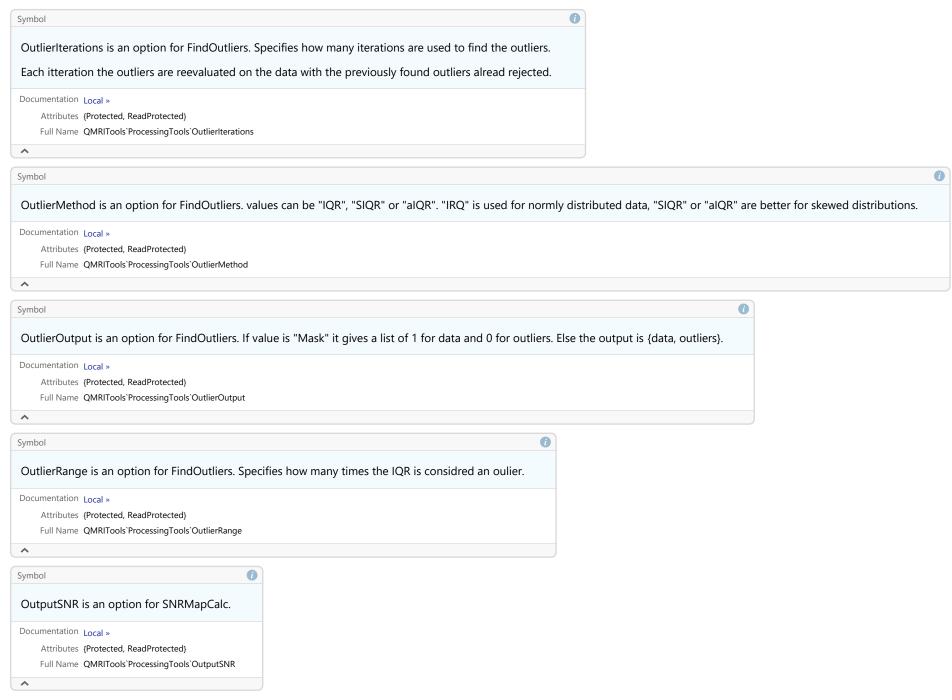


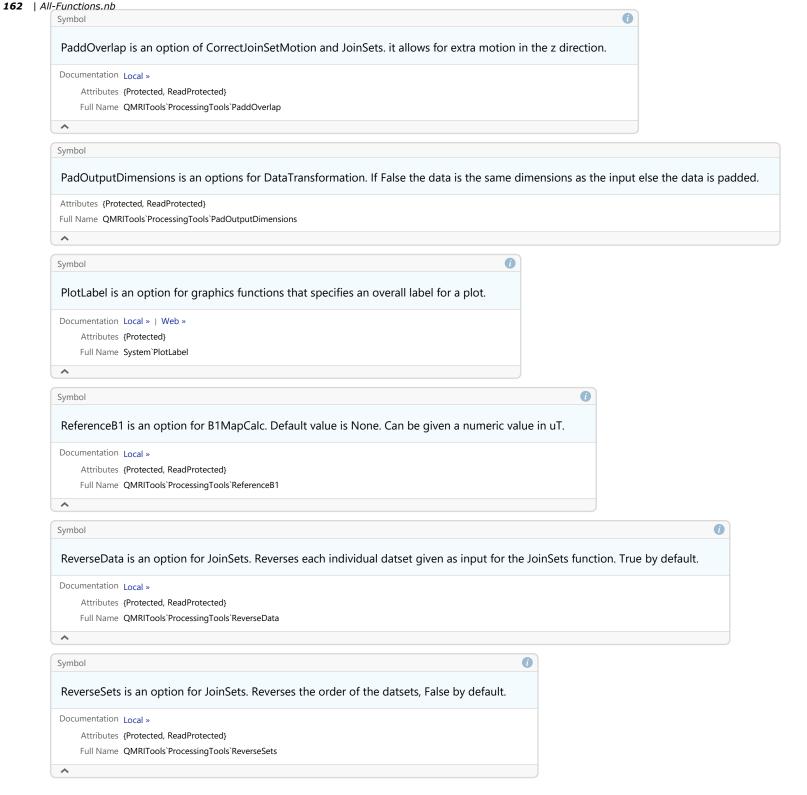


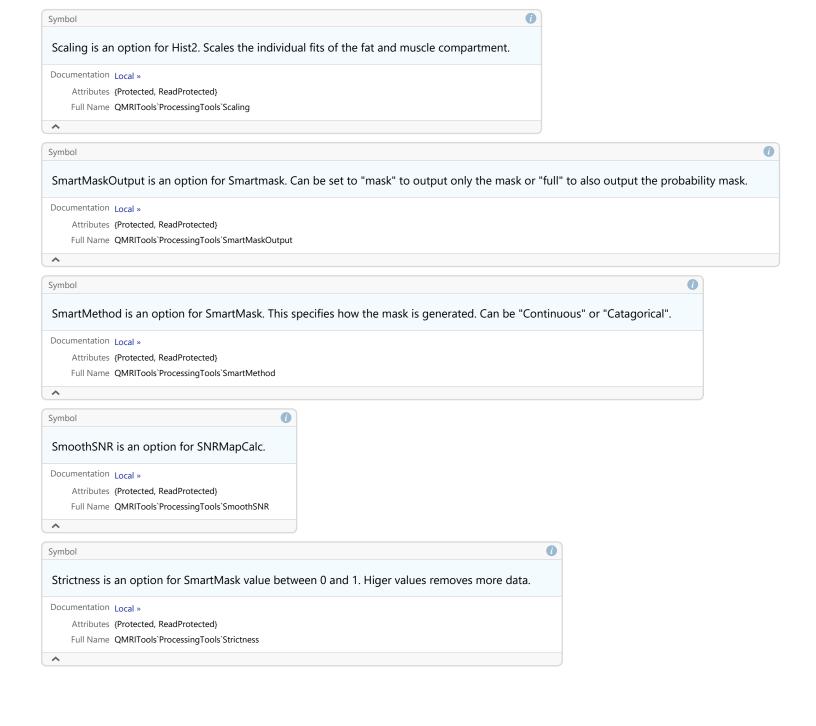


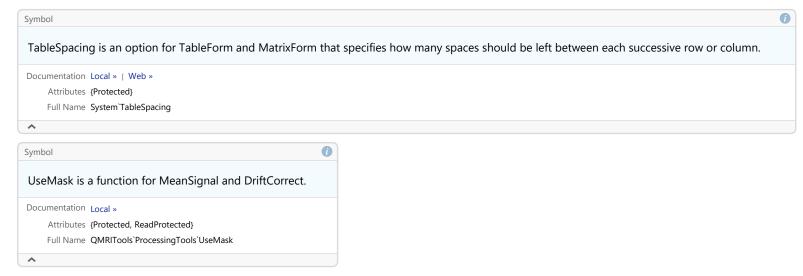






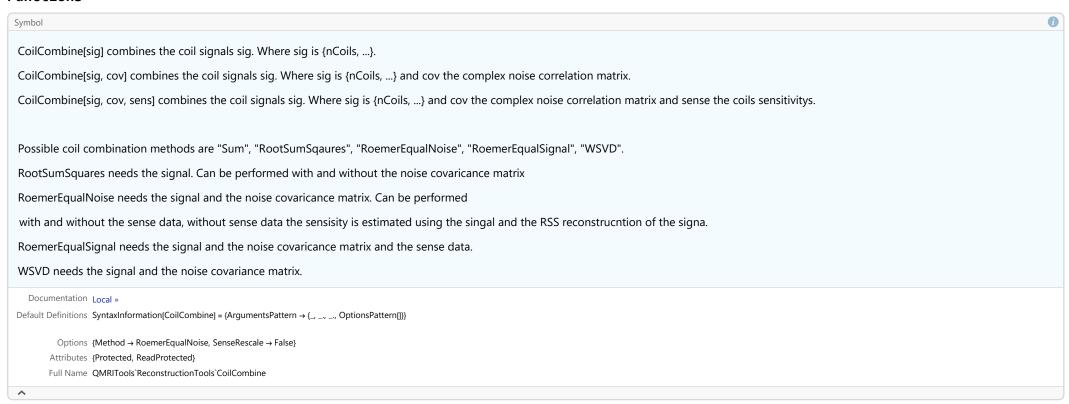




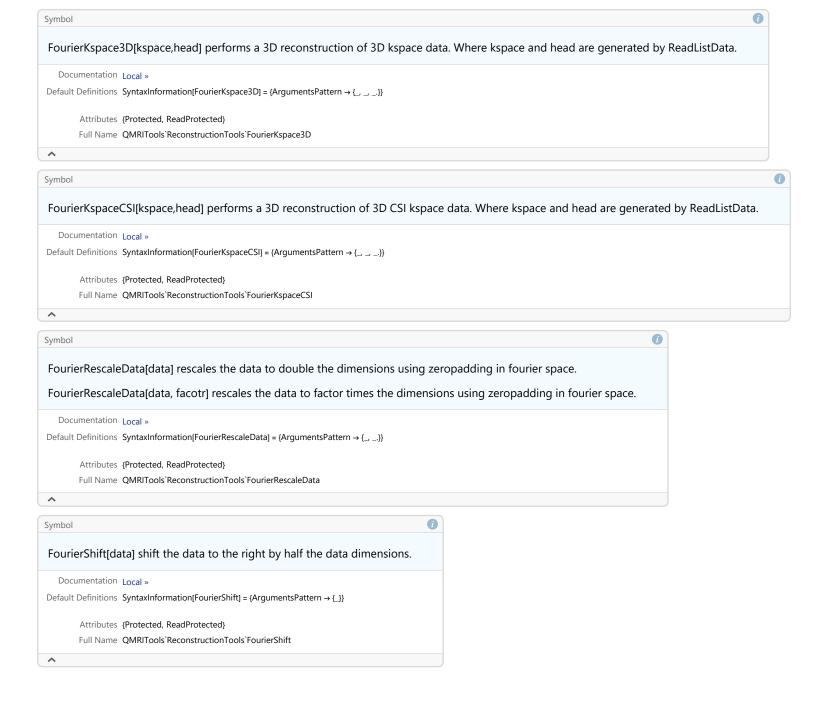


ReconstructionTools

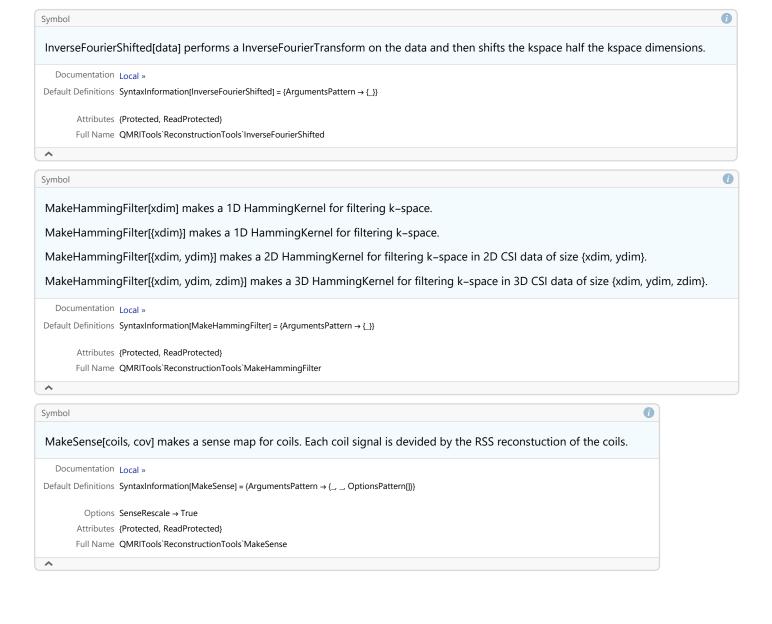
Functions



0

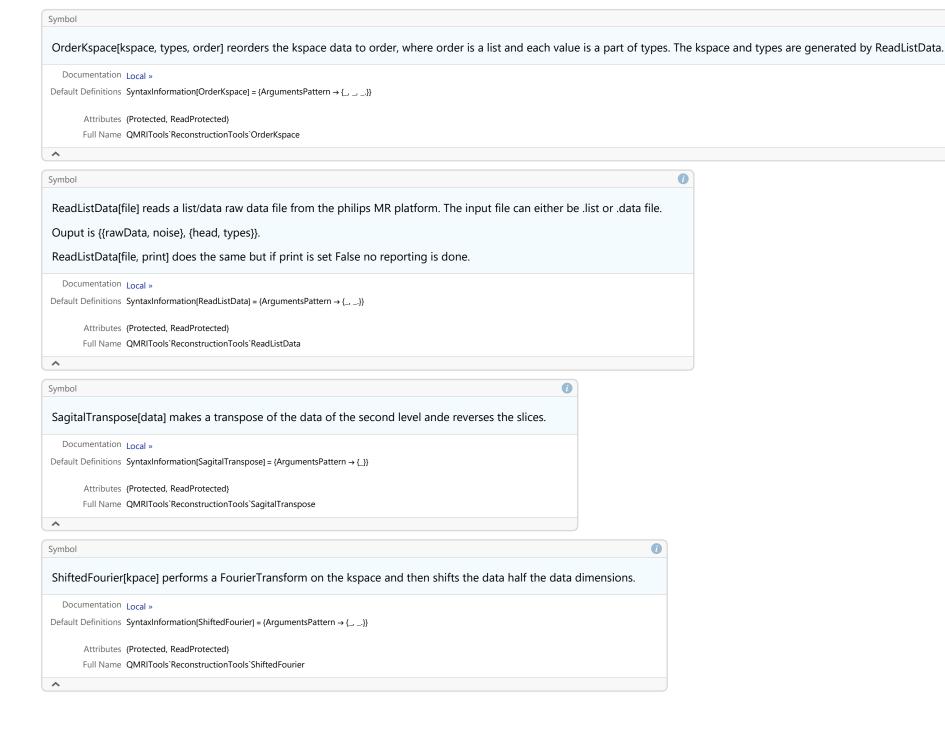


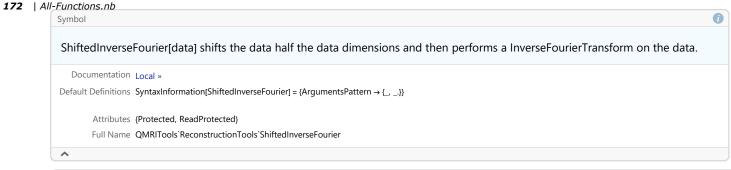
 \wedge

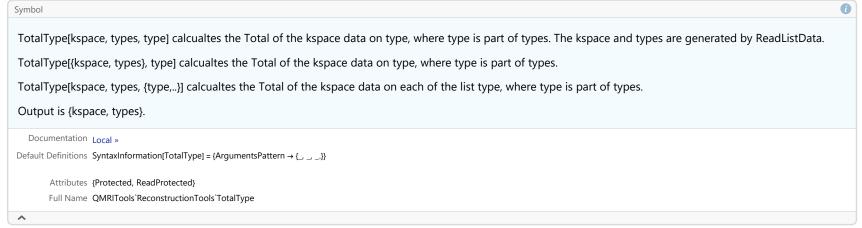


^

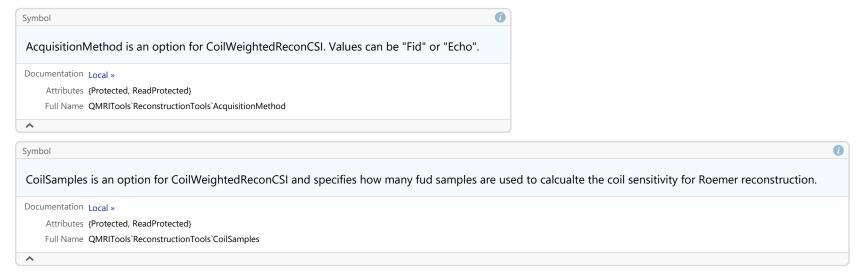
0

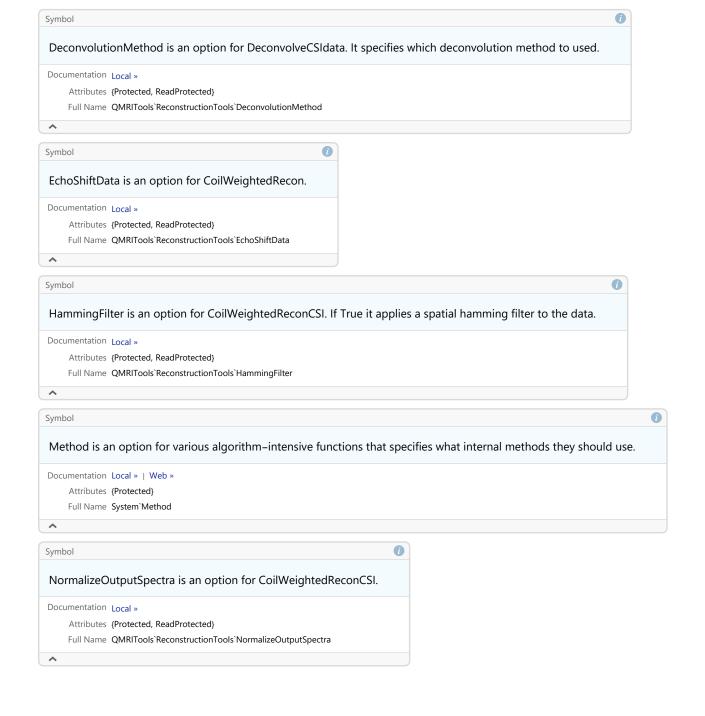


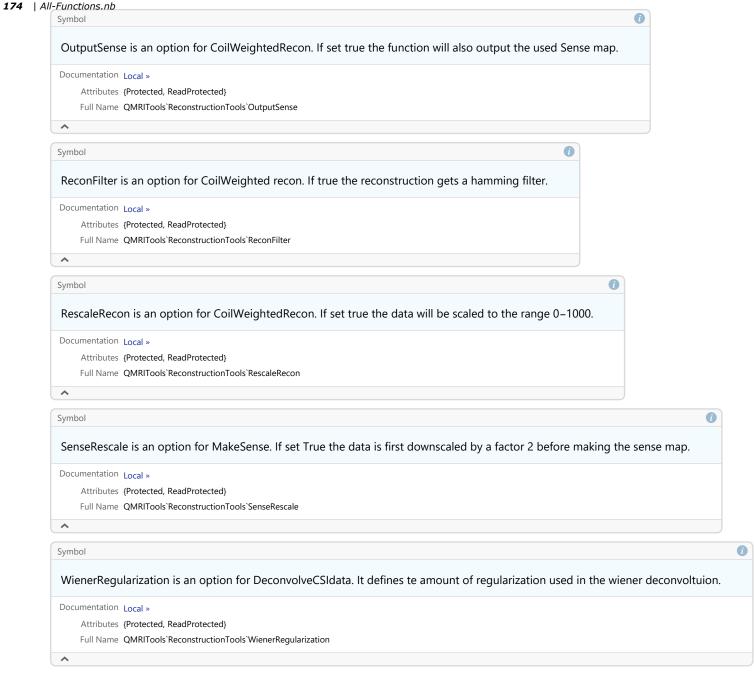




Options







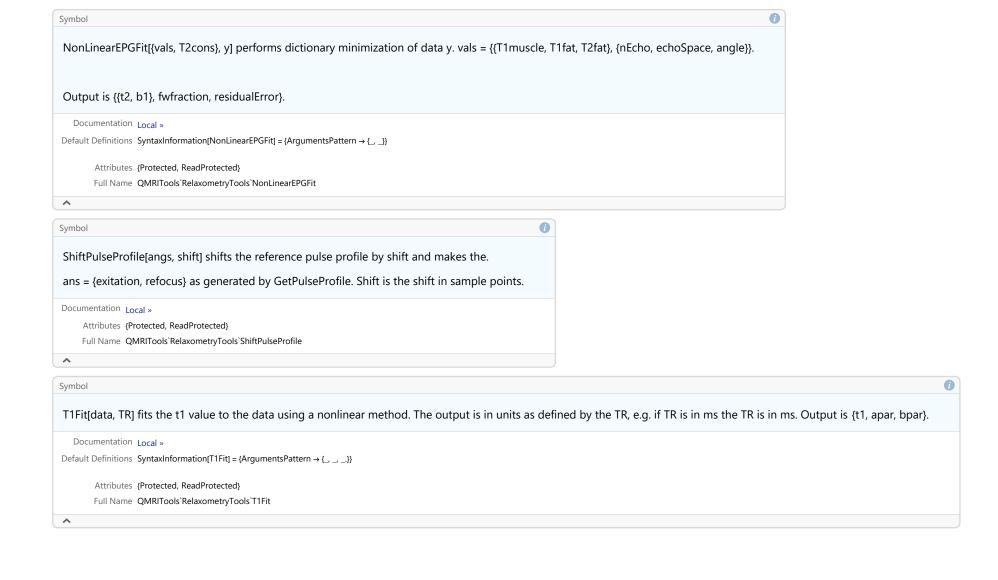
RelaxometryTools

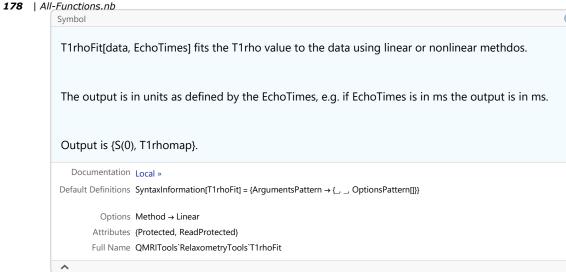
Functions

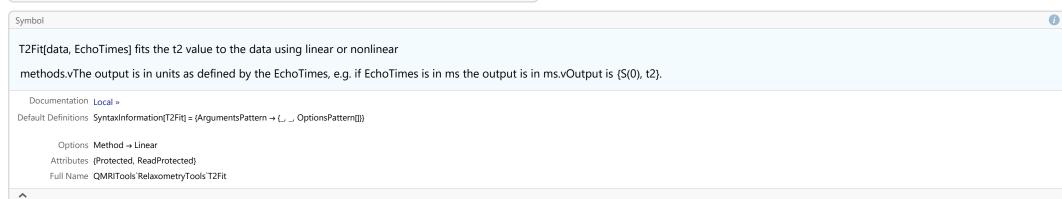
```
Symbol
CalibrateEPGT2Fit[datan, times, angle] calculates the Fat t2 ralaxation that will be used in the EPGT2fit.
Outputs the fat t2 value.
   Documentation Local »
Default Definitions SyntaxInformation[CalibrateEPGT2Fit] = {ArgumentsPattern → {_, _, _, OptionsPattern[]}}
       Options \Rightarrow EPGRelaxPars \rightarrow {{0, 100}, {20, 300}, {1400., 365.}} ... (4 total)
        Attributes {Protected, ReadProtected}
        Full Name QMRITools`RelaxometryTools`CalibrateEPGT2Fit
^
Symbol
CreateT2Dictionary[{T1m, T1f}, {nEcho, detlaTE}, angle] Creates a EPG signal dictionary used for EPGT2fit.
Every dictionary that is defined is cached.
The output is in units as defined by the detlaTE, e.g. if detlaTE is in ms the output is in ms.
The TR and TE should be in the same units as Dela.
Output is {dictionary, vals}.
  Documentation Local »
Default Definitions SyntaxInformation[CreateT2Dictionary] = {ArgumentsPattern → {_, _, _, OptionsPattern[]}}
       Options \rightarrow DictB1Range \rightarrow {0.5, 1.4, 0.01} ... (6 total)
        Attributes {Protected, ReadProtected}
       Full Name QMRITools'RelaxometryTools'CreateT2Dictionary
^
                                                                                                                                                             0
Symbol
DictionaryMinSearch[dictionary, y] performs dictionary minimization of data y. dictionary is generated with CreateT2Dictionary.
Output is {{t2, b1}, fwfraction, residualError}.
   Documentation Local »
Default Definitions SyntaxInformation[DictionaryMinSearch] = \{ArgumentsPattern \rightarrow \{\_, \_, \_, .\}\}
        Attributes {Protected, ReadProtected}
        Full Name QMRITools'RelaxometryTools'DictionaryMinSearch
```

^

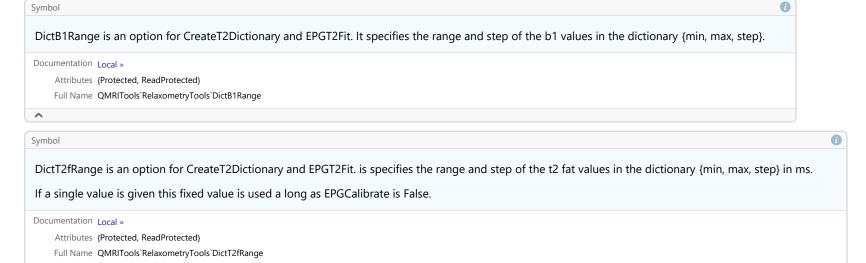
0 Symbol EPGT2Fit[data, {nEcho, detlaTE}, {exitation, refoucs}] fits the t2 based on Marty B et.al. Simultaneous muscle water t2 and fat fraction mapping using transverse relaxometry with stimulated echo compensation. Exitation and refocus are the RF pulse angles e.g. 90,180. They can also be a range of angeles over the slice profile as defined by GetSliceProfile. The output is in units as defined by the detlaTE, e.g. if detlaTE is in ms the output is in ms. The exitation and refocus are defined in Degrees. Output is {{{T2map,B1Map},{wat, fat, fatMap}, residual}, callibration} or {{T2map,B1Map},{wat, fat, fatMap}, residual}. EPGT2Fit[] is based on DOI: 10.1002/nbm.3459. Documentation Local » Default Definitions SyntaxInformation[EPGT2Fit] = {ArgumentsPattern → {_, _, _, OptionsPattern[]}} Options **>** EPGRelaxPars → {1400., 365.} ... (17 total) Attributes {Protected, ReadProtected} Full Name QMRITools'RelaxometryTools'EPGT2Fit ^

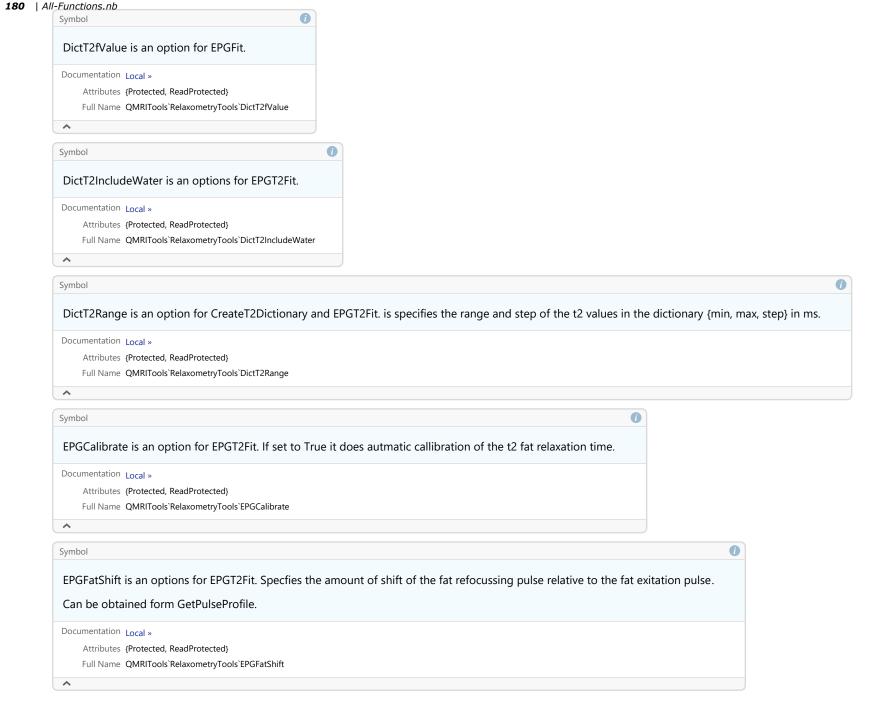


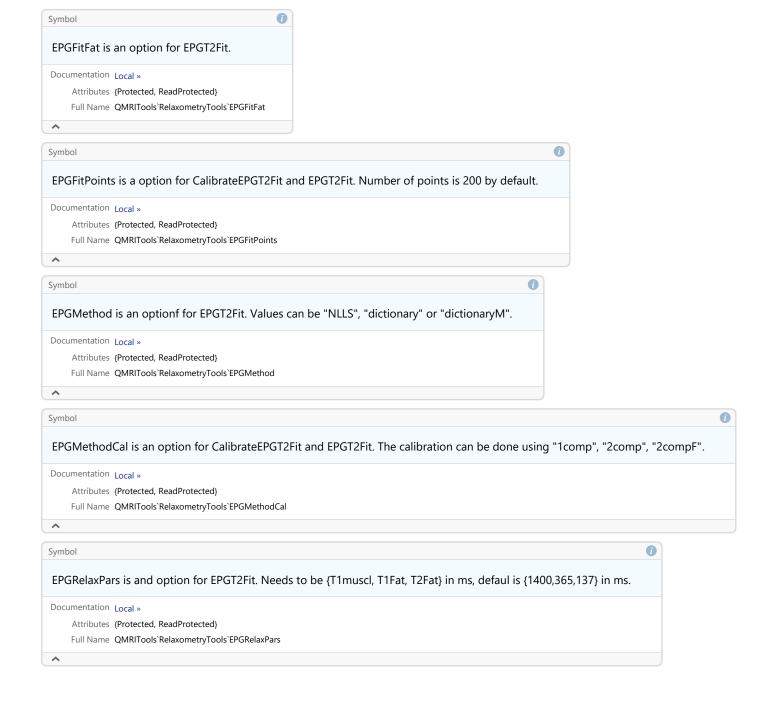


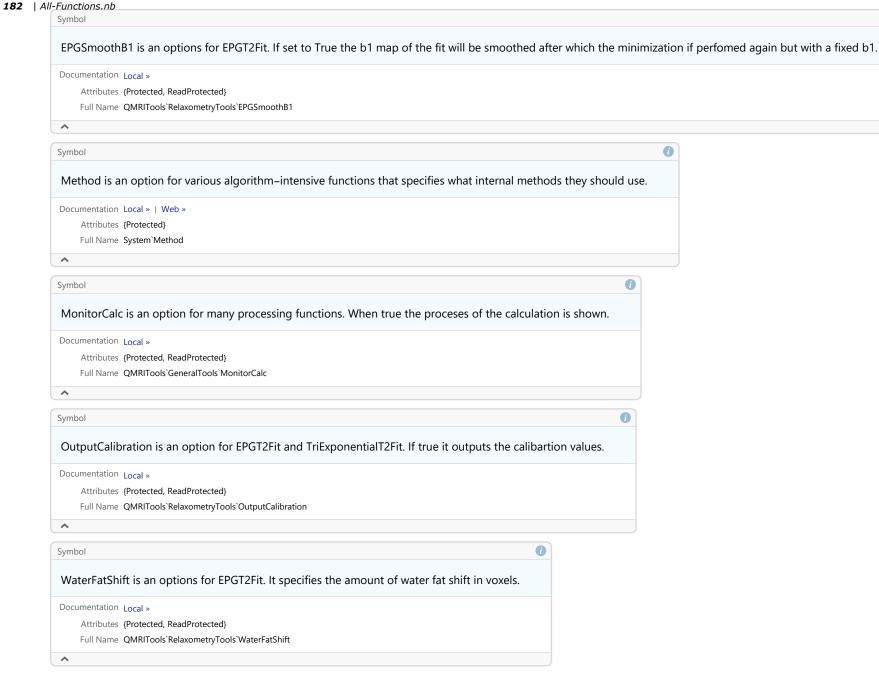


Options









```
WaterFatShiftDirection is an options for EPGT2Fit. It specifies the water fat shift direction: "left", "right", "up" and "down".

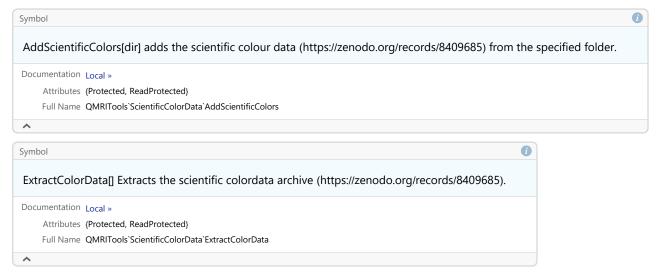
Documentation Local »

Attributes {Protected, ReadProtected}

Full Name QMRITools'RelaxometryTools'WaterFatShiftDirection
```

ScientificColorData

Functions



Options

SegmentationTools

Functions



^

Full Name QMRITools'SegmentationTools'DiceLossLayer

 \wedge

0 Symbol DiceSimilarity[ref, pred] gives the Dice Similarity between 1 and 0 of segmentations ref and pred for class equals 1. DiceSimilarity[x, y, class] gives the Dice Similarity of segmentations ref and pred for class. DiceSimilarity[x, y, {class, ..}] gives the Dice Similarity of segmentations ref and pred for the list of gives classes. Documentation Local » Default Definitions SyntaxInformation[DiceSimilarity] = {ArgumentsPattern → {_, _, _, ...}} Attributes {Protected, ReadProtected} Full Name QMRITools'SegmentationTools'DiceSimilarity ^

Symbol

FocalLossLayer[] represents a net layer that computes the Focal loss by comparing input class probability vectors with the target class vector.

0

FocalLossLayer[q] does the same but uses q as the tunable focusing parameter gamma which needs to be larger than one.

FocalLossLayer[g, a] does the same but uses as the balancing factor alpha.

Default Definitions SyntaxInformation[FocalLossLayer] = {ArgumentsPattern $\rightarrow \{_, _.\}$ } Attributes {Protected, ReadProtected}

Full Name QMRITools`SegmentationTools`FocalLossLayer

Symbol

^

^

GetNeuralNet[name] loads a pretrained neural net that come with the toolbox. Current named nets

are "LegSide", "LegSide", "SegThighMuscle", "SegLegMuscle", and "SegLegBones". The loading is cashed within a session.

Documentation Local »

Default Definitions SyntaxInformation[GetNeuralNet] = {ArgumentsPattern \rightarrow {_}}

Attributes {Protected, ReadProtected}

Full Name QMRITools`SegmentationTools`GetNeuralNet

GetTrainData[data, batchsize, patch] creates a training batch of size batchsize with patchsize patch.

The input data can be out of memory in the form of a list of "*wxf" files that contain the data, segmentation and voxel size or a list of "*.nii" files in the form {{"data.nii", "segmentation.nii"}...}. The input data can be in memory in a list in the form {{data, segmentation, vox}...}

0

 $GetTrainData[data, batchsize, patch, nClass] \ If \ nClass \ is \ set \ to \ an \ value \ n > 0 \ the \ segmentations \ are \ decoded \ in \ n \ classes.$

Documentation Local »

Default Definitions SyntaxInformation[GetTrainData] = {ArgumentsPattern → { , , , , , , optionsPattern[]}}

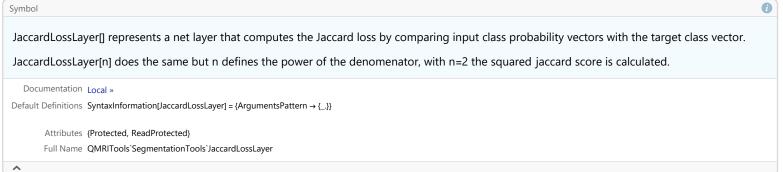
Options {PatchesPerSet → 1, AugmentData → True}

Attributes {Protected, ReadProtected}

Full Name QMRITools'SegmentationTools'GetTrainData



^



0

JaccardSimilarity[ref, pred] gives the Jaccard Similarity between 1 and 0 of segmentations ref and pred for class equals 1.

JaccardSimilarity[x, y, class] gives the Jaccard Similarity of segmentations ref and pred for class.

JaccardSimilarity[x, y, {class, ..}] gives the Jaccard Similarity of segmentations ref and pred for the list of gives classes.

Documentation

Local >>

Default Definitions SyntaxInformation[JaccardSimilarity] = {ArgumentsPattern → {., ., ...}}

Attributes {Protected, ReadProtected}

Full Name QMRITools'SegmentationTools'JaccardSimilarity

Symbol

MakeChannelClassGrid[data, label] makes a 3 x 3 grid of crossectional images of the channels data overlaid with a crossectional image of the classes label of a training dataset generated MakeChannelClassGrid[data, label, n] makes a n x n.

MakeChannelClassGrid[data, label, {n, m}] makes a n x m.

Default Definitions SyntaxInformation[MakeChannelClassGrid] = {ArgumentsPattern → {_, _, _, _.}}

Attributes {Protected, ReadProtected}

Full Name QMRITools`SegmentationTools`MakeChannelClassGrid

Symbol

^

MakeChannelClassImage[data, label] makes a crossectional image of the channels data overlaid with a crossectional image of the classes label of a training dataset generated MakeChannelClassImage[data, label, {off,max}] same but with explicit definition of background value b and number of classes n.

MakeChannelClassImage[data, label, vox] same but with the aspect ratio determined by vox.

MakeChannelClassImage[data, label, {off,max}, vox] same with explicit definition and aspect ratio definition.

Documentation Local »

Default Definitions SyntaxInformation[MakeChannelClassImage] = $\{ArgumentsPattern \rightarrow \{_, _, _, \}\}$

Attributes {Protected, ReadProtected}

Full Name QMRITools`SegmentationTools`MakeChannelClassImage

^

MakeClassImage[label] makes a crossectional image of the classes label of a training dataset generated by GetTrainData

MakeChannelImage[label, {b, n}] same but with explicit definition of background value b and number of classes n.

MakeClassImage[data, vox] same but with the aspect ratio determined by vox.

MakeChannelImage[label, {b, n}, vox] same with explicit definition and aspect ratio definition.

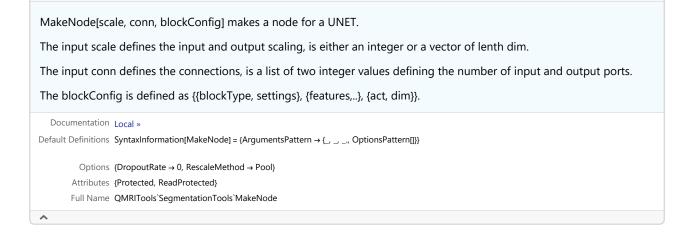
Documentation Local »

Default Definitions SyntaxInformation[MakeClassImage] = {ArgumentsPattern → { , _ _ , _ _}}}

Attributes {Protected, ReadProtected}
Full Name QMRITools'SegmentationTools'MakeClassImage

MakeDistanceMap[mask] makes a distance map of the given mask in voxels. The distance map is negative inside the mask and positive outside the mask. MakeDistanceMap[mask, vox] makes a distance map of the given mask in the same unit as vox. The distance map is negative inside the mask and positive outside the mask. Default Definitions SyntaxInformation[MakeDistanceMap] = {ArgumentsPattern → { __ _ _ , OptionsPattern[]}} Options DistanceRange → Automatic Attributes {Protected, ReadProtected} Full Name QMRITools SegmentationTools MakeDistanceMap

0



^

Symbol

TverskyLossLayer[] represents a net layer that computes the Tversky loss by comparing input class probability vectors with the target class vector.

TverskyLossLayer[b] does the same but b defines the tversky beta factor. With beta = 0.5 its is the Dice coefficient. Here alpha + beta = 1.

Documentation Local »

Default Definitions SyntaxInformation[TverskyLossLayer] = {ArgumentsPattern → {..}}

Attributes {Protected, ReadProtected}
Full Name QMRITools'SegmentationTools'TverskyLossLayer

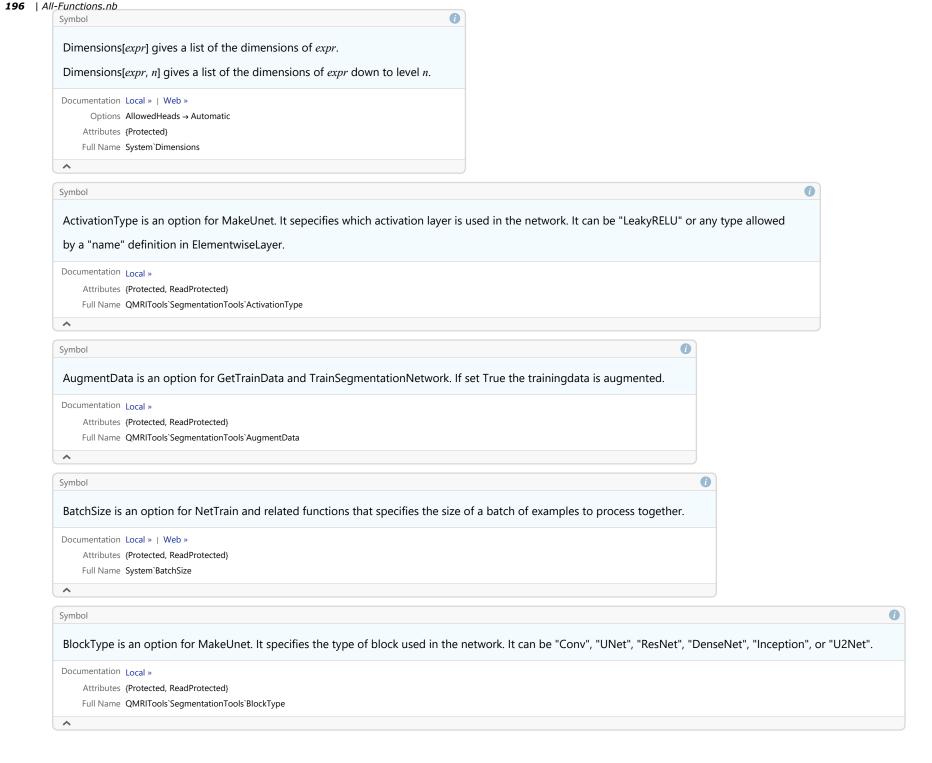
QMRITools`SegmentationTools`\$debugUnet

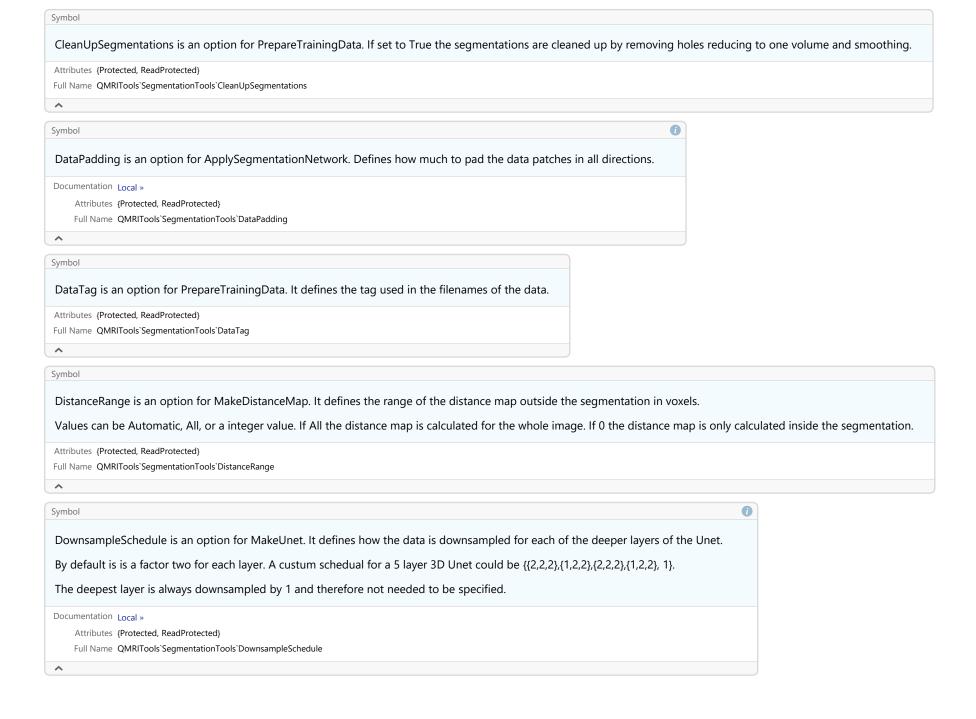
Attributes {ReadProtected}
Full Name QMRITools`SegmentationTools`\$debugUnet

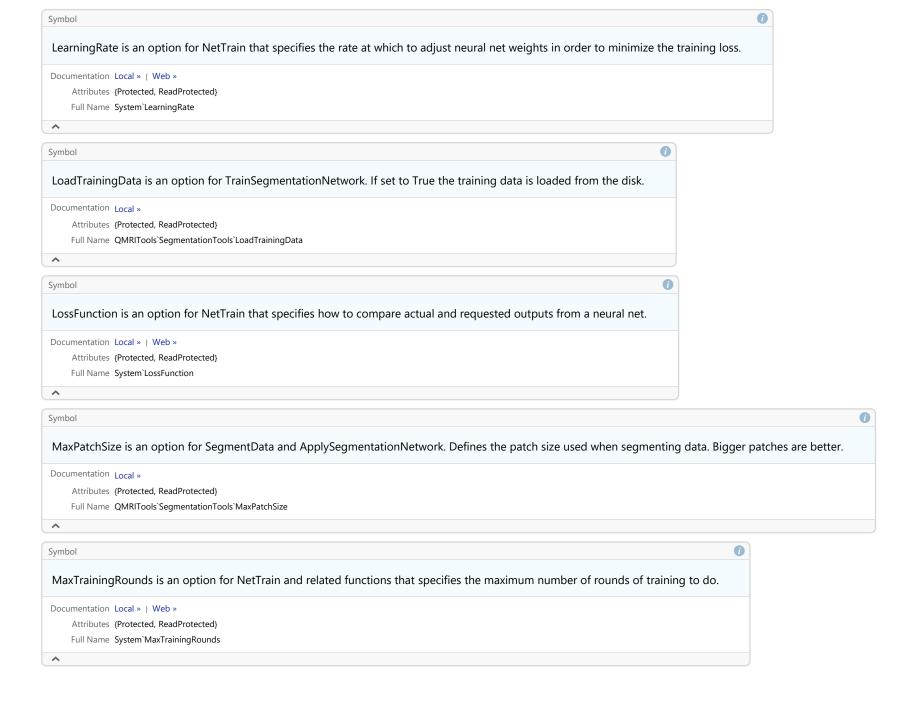
Options

Missing[UnknownSymbol, Channels]

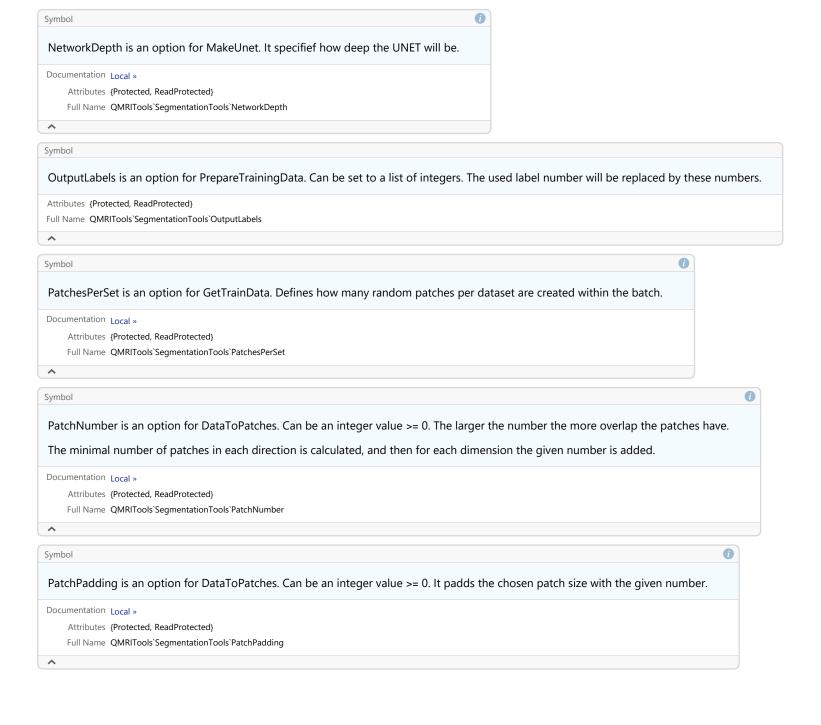
Missing[UnknownSymbol, Classes]

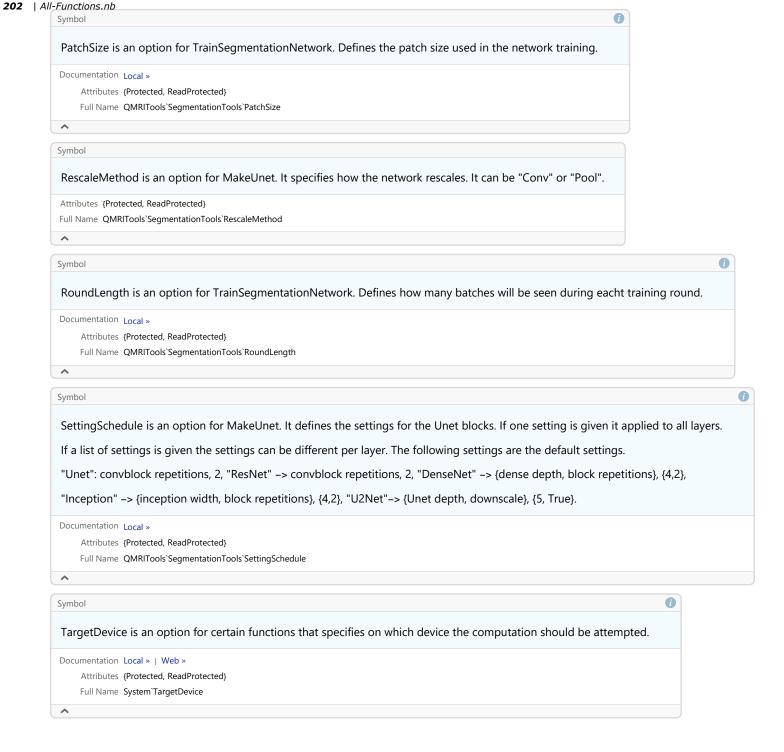






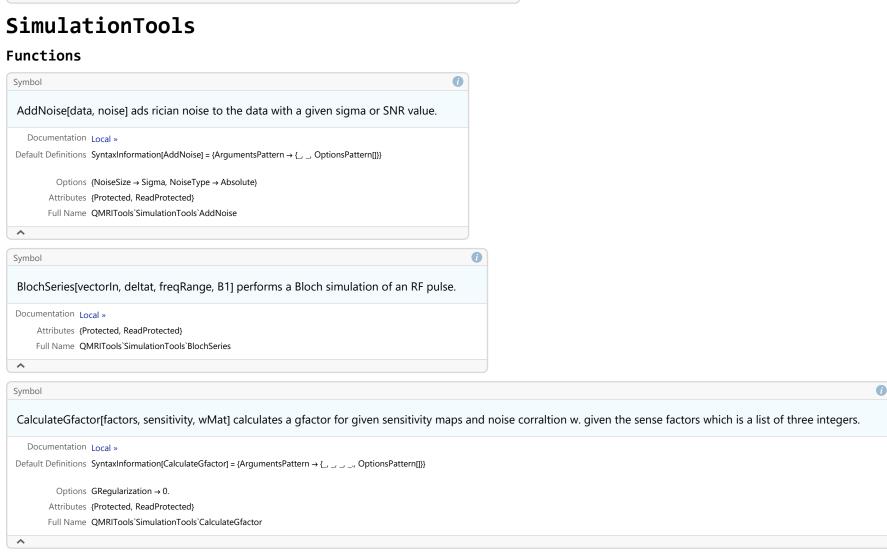
 \wedge





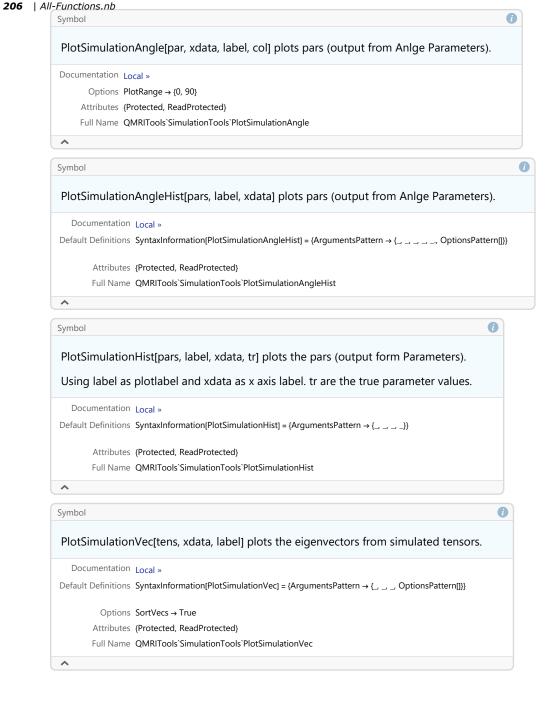
```
TestRun is an option for PrepareTrainingData. If set to True the data is not saved only analyzed.

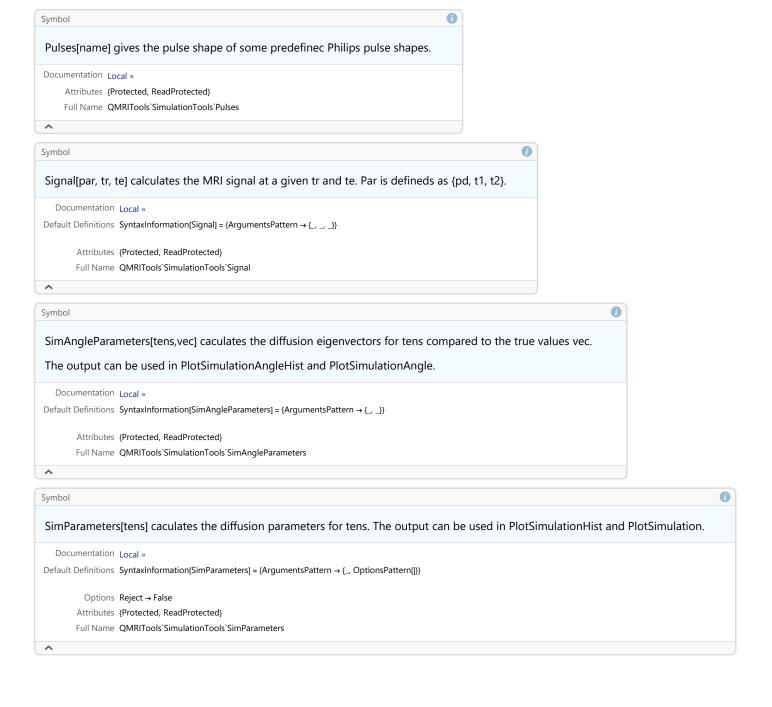
Attributes {Protected, ReadProtected}
Full Name QMRITools`SegmentationTools`TestRun
```

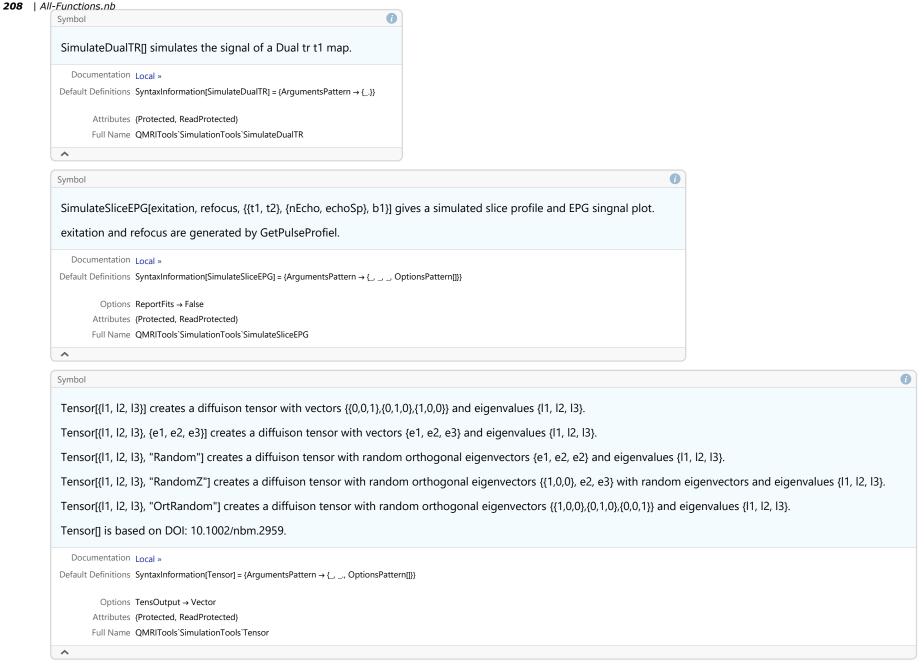


 \wedge

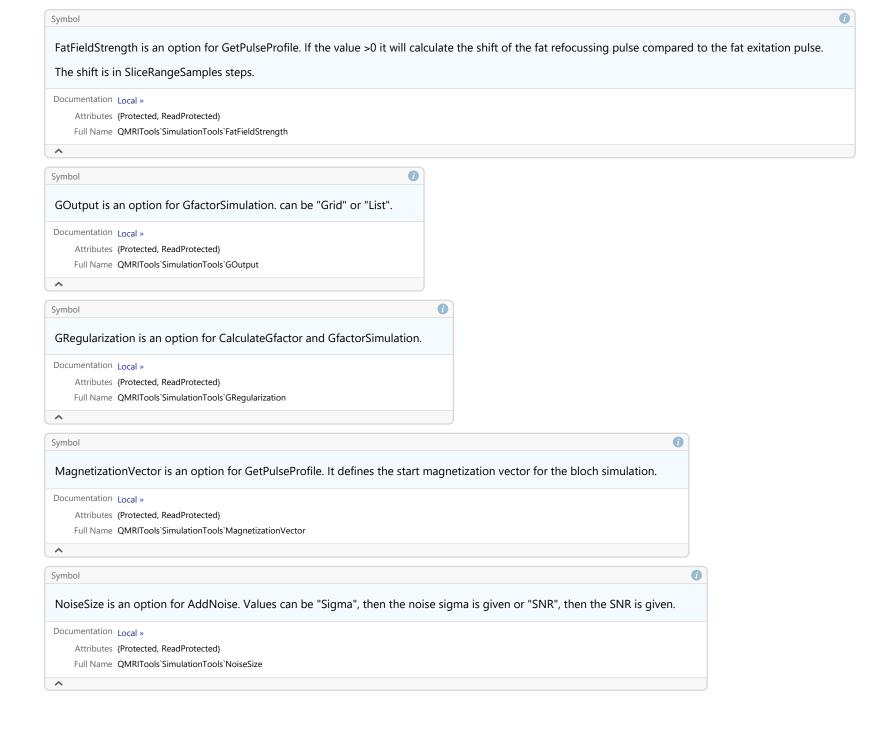
```
Symbol
GetPulseProfile[excitation, refocus] gives the pusl angle profiles for the exitation and refocussing pulses.
a pulse is defined as {"name", flipangle, {G_strnth, Dur, BW}}.
GetPulseProfile[{"name", flipangle, {G_strnth, Dur, BW}}] gives detaile slice profile information of one pulse.
output is {ex_angle_profiel, ref_angel_profile, {plots}}.
output for single pulse is {{distance, Mt, Mz, Mx, My, ang, phase}, plots}.
  Documentation Local »
Default Definitions SyntaxInformation[GetPulseProfile] = {ArgumentsPattern → {_, _, _, OptionsPattern[]}}
       Options ➤ MagnetizationVector → {0, 0, 1} ... (4 total)
        Attributes {Protected, ReadProtected}
       Full Name QMRITools`SimulationTools`GetPulseProfile
\wedge
Symbol
GfactorSimulation[sensitivity, cov, {dir,sense}] calculates the gfactormaps for given sensitivity maps and noise corraltion cov in one direction.
The sensefactors are a list of integers in a given direction: "LR", "FH", or "AP".
GfactorSimulation[sensitivity, cov, {dir1,sense1}, {dir2,sense2}] calculates the gfactormaps for given sensitivity maps and noise corraltion w in two directions.
  Documentation Local »
Default\ Definitions\ \ SyntaxInformation[GfactorSimulation] = \{ArgumentsPattern \rightarrow \{\_,\_,\_,\_,OptionsPattern[]\}\}
         Options {GRegularization → 0., GOutput → Grid}
        Attributes {Protected, ReadProtected}
       Full Name QMRITools'SimulationTools'GfactorSimulation
^
Symbol
PlotSimulation[pars, xval, true, label, color] plots the pars (output form Parameters). Using label as PlotLabel and xval as x axis Thics.
tr are the true parameter values. color are the color used for the plot.
  Documentation Local »
Default Definitions SyntaxInformation[PlotSimulation] = {ArgumentsPattern → {_, _, _, _, _, OptionsPattern[]}}
         Options PlotRange \rightarrow {{0, 3}, {0, 3}, {0, 3}, {0, 1}}
        Attributes {Protected, ReadProtected}
       Full Name QMRITools`SimulationTools`PlotSimulation
\wedge
```

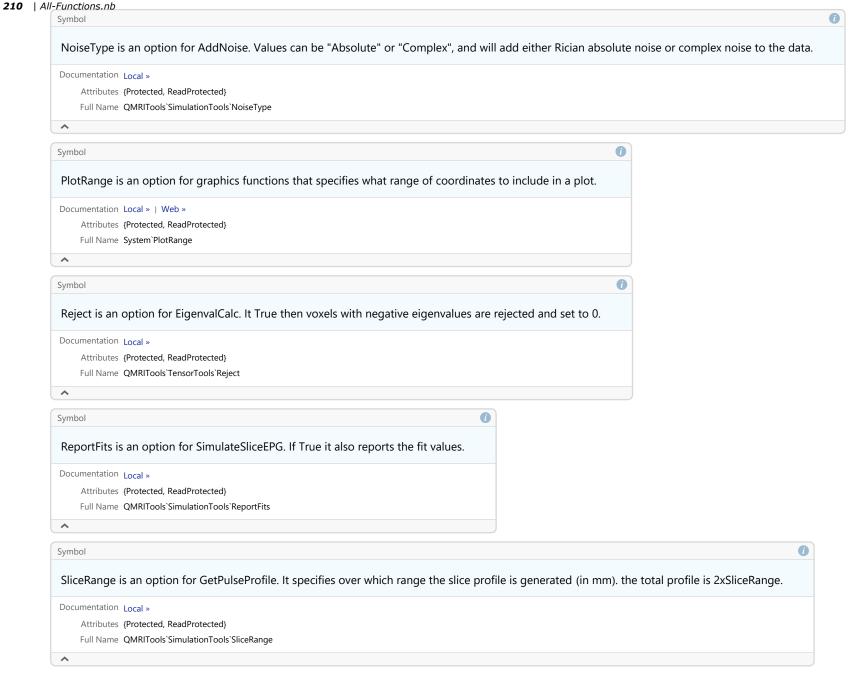


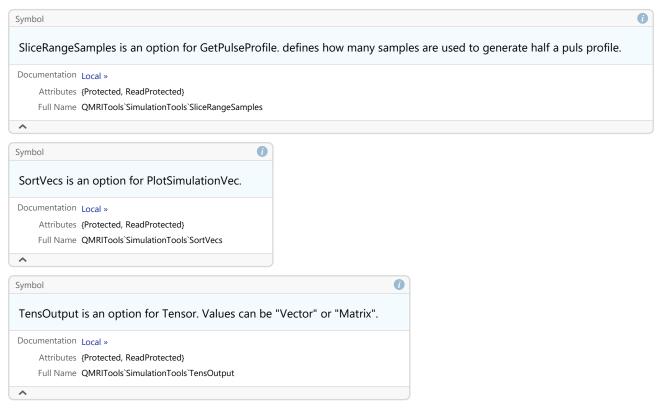




Options

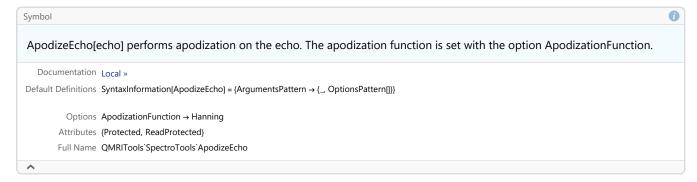


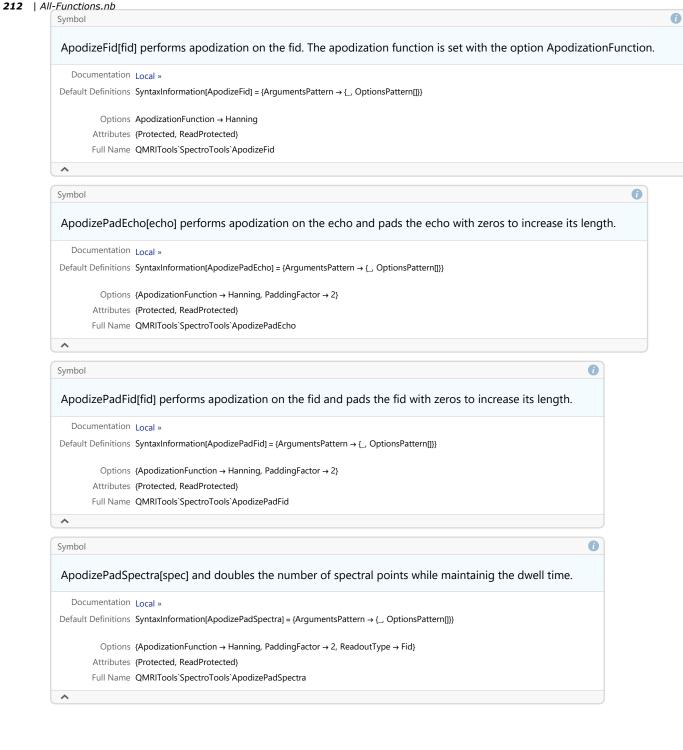


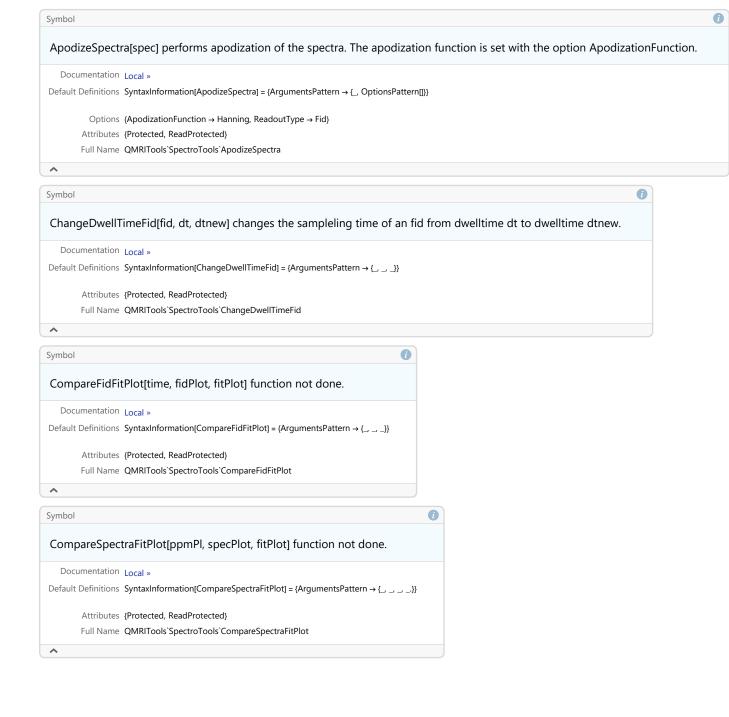


SpectroTools

Functions







CSIInterface[] opens the CSI interface. Function not done.

CSIInterface[te, bw] opens the CSI interface with known te and bw.

CSIInterface[file] opens the CSI interface with the data from file loaded.

CSIInterface[file, {tei, bwi}] opens the CSI interface with the data from file loaded with known te and bw.

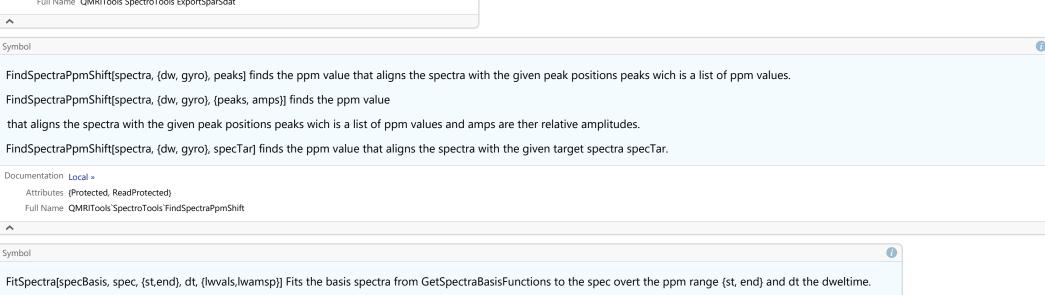
Documentation Local »

Default Definitions SyntaxInformation[CSIInterface] = {ArgumentsPattern → {_,, _,, optionsPattern[]}}

Options {SpectraFieldStrength → 7, SpectraNucleus → 31P}

Attributes {Protected, ReadProtected}

Full Name QMRITools'SpectroTools'CSIInterface



FitSpectra[specBasis, spec, {st,end}, dt, {lwvals,lwamsp}] Fits the basis spectra from GetSpectraBasisFunctions to the spec overt the ppm range {st, end} and dt the dweltime.

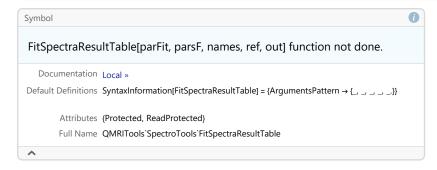
Documentation Local »

Default Definitions SyntaxInformation[FitSpectra] = {ArgumentsPattern → { _ _ _ _ _ _ _ _ OptionsPattern[]}}

Options ➤ SpectraNucleus → 1H ... (10 total)

Attributes {Protected, ReadProtected}

Full Name QMRITools'SpectroTools'FitSpectra

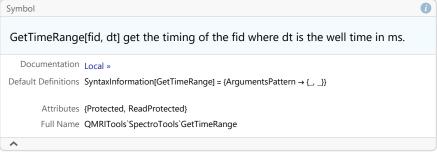


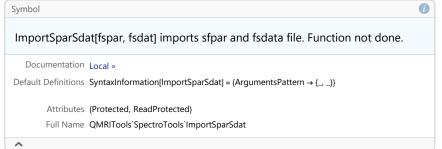
Options **>** BasisSequence → {PulseAcquire, 0} ... (6 total)

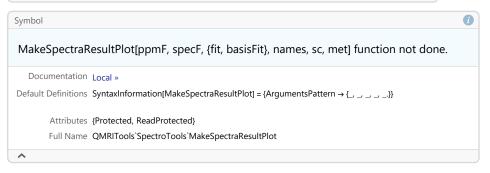
Full Name QMRITools`SpectroTools`GetSpectraBasisFunctions

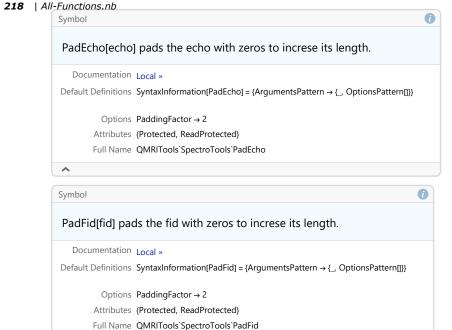
Attributes {Protected, ReadProtected}

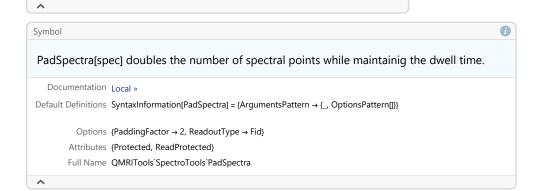
^











0

0

PhaseCorrectSpectra[spec] performs 0th order phase correction of the spectra by minimizing the difference between the real and absolute spectra velaue.

PhaseCorrectSpectra[spec, dw] performs 0th order phase correction of the spectra using Henkel matrix SVD fitting.

PhaseCorrectSpectra[spec, dw, te] := performs 0th and 1st order phase correction of the spectra

using Henkel matrix SVD fitting. The first order phase is corrected by padding the fid with the missing values in the time befroe the TE.

PhaseCorrectSpectra[spec, dw, te, gyro, ppmRan] performs 0th and 1st order phase

correction of the spectra using Henkel matrix SVD fitting. Only the part of the spectra in the ppmRan is used for optimization.

Documentation Local »

Attributes {Protected, ReadProtected}

Full Name QMRITools`SpectroTools`PhaseCorrectSpectra

^

Symbol

PhaseShiftSpectra[spectra, phi0] aplies the 0th order phase phi0 to the spectra.

PhaseShiftSpectra[spectra, ppm, gyro, phi1] aplies the 1st order phase phi1 to the spectra. The ppm can be obtained using GetPpmRange and gyro with GetGyro.

PhaseShiftSpectra[spec, ppm, gyro, {phi0, phi1}] aplies the 0th and 1st order phases {phi0, phi1} to the spectra. The ppm can be obtained using GetPpmRange and gyro with GetGyro.

The 0th order phase phi0 is in radians and the 1st order phase phi1 is in ms.

Documentation Local »

Default Definitions SyntaxInformation[PhaseShiftSpectra] = {ArgumentsPattern $\rightarrow \{_, _, _, _\}$ }

Attributes {Protected, ReadProtected}

Full Name QMRITools`SpectroTools`PhaseShiftSpectra

^

Symbol

PlotCSIData[spectra, {dwell, gyro}] plots the CSI spectra which has dimensions {z,y,x,nsamp}. The ppm axes is determined by dwell and gyro. Gyro can be obtained with GetGyro.

PlotCSIData[spectra, {dwell, field, nuc}] plots the CSI spectra which has dimensions {z,y,x,nsamp}. The ppm axes is determined by dwell and field and nuc.

Documentation Local »

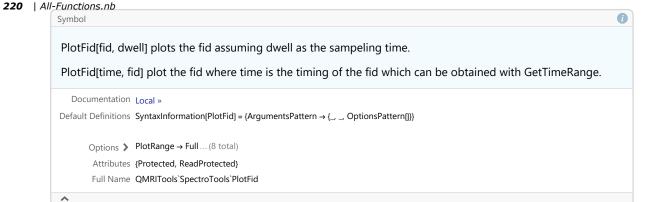
Default Definitions SyntaxInformation[PlotCSIData] = {ArgumentsPattern → {______, OptionsPattern[]}

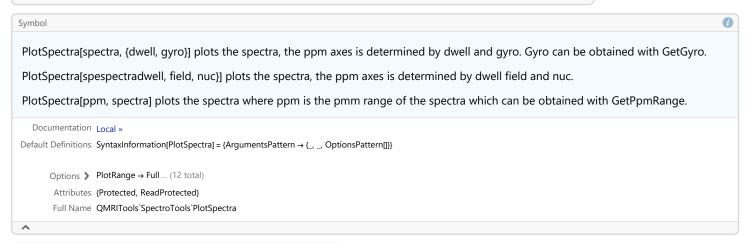
Options PlotRange → Full

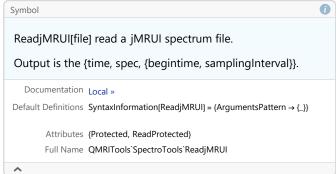
Attributes {Protected, ReadProtected}

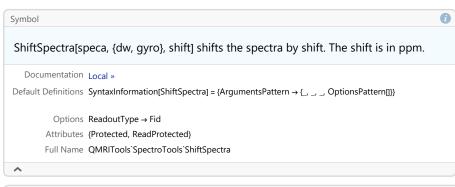
Full Name QMRITools`SpectroTools`PlotCSIData

^











Symbol

TimeShiftEcho[fid, time, gam] aplies a linebroadening with linewidth gam and a Voigt lineshape to the fid. The time can be obtained using GetTimeRange.

TTimeShiftEcho[fid, time, {gam, f}] aplies a linebroadening with linewidth gam and a custom lineshape f to the fid (f=0, "Gaussinan", f=1 "Lorentzian").

TTimeShiftEcho[fid, time, gyro, {gam, eps}] aplies a linebroadening with linewidth gam to the fid and a phase eps that results in eps ppm shift of the spectra. The gyro can be obtained with GetGyro.

TTimeShiftEcho[fid, time, gyro, {gam, eps, f}] aplies a linebroadening with linewidth gam using a custom lineshape f to the fid and a phase eps that results in eps ppm shift of the spectra.

The linewidth gam is given in ms and the spectra shift eps is given in ppm.

```
Documentation Local »

Default Definitions SyntaxInformation[TimeShiftEcho] = {ArgumentsPattern → { , , , , , ...}}

Attributes {Protected, ReadProtected}

Full Name QMRITools`SpectroTools`TimeShiftEcho
```

TimeShiftFid[fid, time, gam] aplies a linebroadening with linewidth gam and a Voigt lineshape to the fid. The time can be obtained using GetTimeRange.

TimeShiftFid[fid, time, {gam, f}] aplies a linebroadening with linewidth gam and a custom lineshape f to the fid (f=0, "Gaussian", f=1 "Lorentzian").

TimeShiftFid[fid, time, gyro, {gam, eps}] aplies a linebroadening with linewidth gam to the fid and a phase eps that results in eps ppm shift of the spectra. The gyro can be obtained with GetGyro.

TimeShiftFid[fid, time, gyro, {gam, eps, f}] aplies a linebroadening with linewidth gam using a custom lineshape f to the fid and a phase eps that results in eps ppm shift of the spectra.

The linewidth gam is given in ms and the spectra shift eps is given in ppm.

```
Documentation Local »

Default Definitions SyntaxInformation[TimeShiftFid] = {ArgumentsPattern → { , , , , , , , }}}

Attributes {Protected, ReadProtected}

Full Name QMRITools`SpectroTools`TimeShiftFid
```

Symbol

TimeShiftFidV[fid, time, gam] aplies a linebroadening with linewidth gam and a Voigt lineshape to the fid. The time can be obtained using GetTimeRange.

TimeShiftFidV[fid, time, {gamL, gamG}] aplies a linebroadening with linewidth gamG "Gaussian" and gamL "Lorentzian".

TimeShiftFidV[fid, time, gyro, {gam, eps}] aplies a linebroadening with linewidth gam to the fid and a phase eps that results in eps ppm shift of the spectra. The gyro can be obtained with GetGyro.

0

TimeShiftFidV[fid, time, gyro, {{gamL, gamG}, eps}] aplies a linebroadening with

linewidth linewidth gamG "Gaussian" and gamL "Lorentzian" to the fid and a phase eps that results in eps ppm shift of the spectra.

The linewidth gam is given in ms and the spectra shift eps is given in ppm.

```
Documentation Local »

Default Definitions SyntaxInformation[TimeShiftFidV] = {ArgumentsPattern → { _ _ _ _ _ _ _ }}

Attributes {Protected, ReadProtected}

Full Name QMRITools`SpectroTools`TimeShiftFidV
```

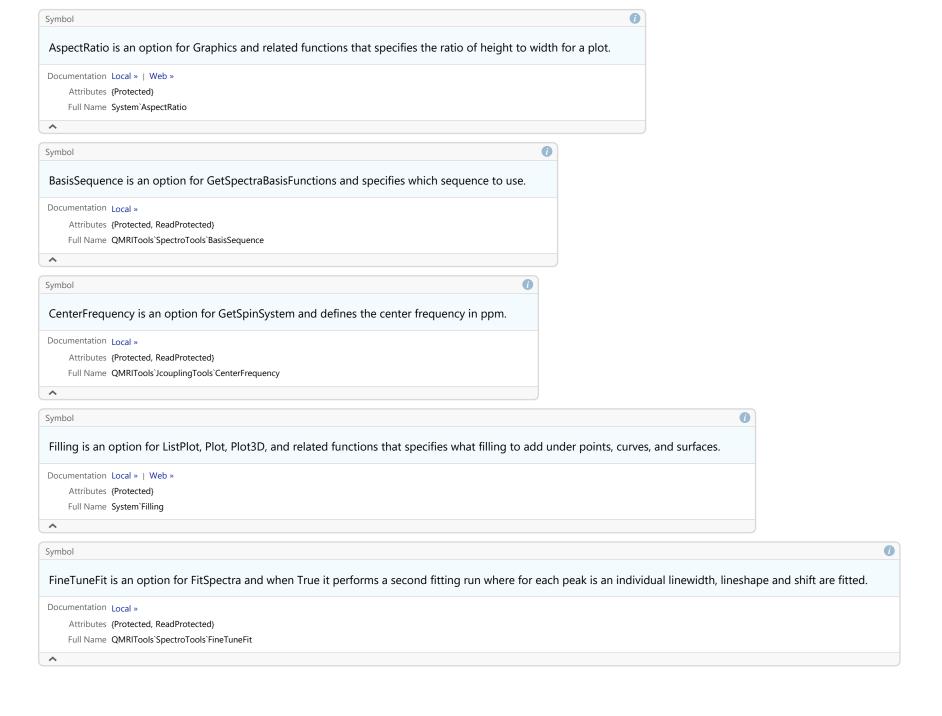
Options

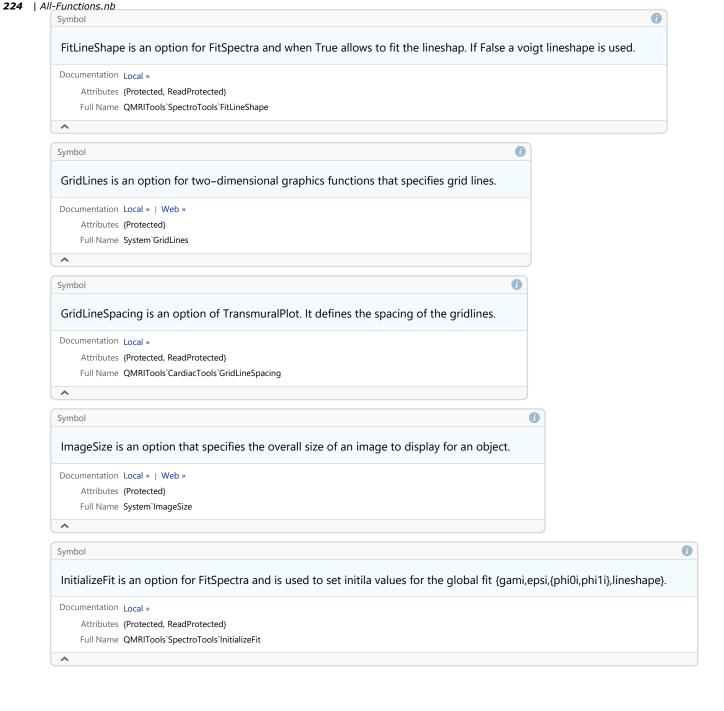
ApodizationFunction is an options for ApodizeFid, ApodizeSpectra, ApodizePadFid, and ApodizePadSpectra. Values can be "Hanning", "Gaussian", "Lorentzian", and "Voigt".

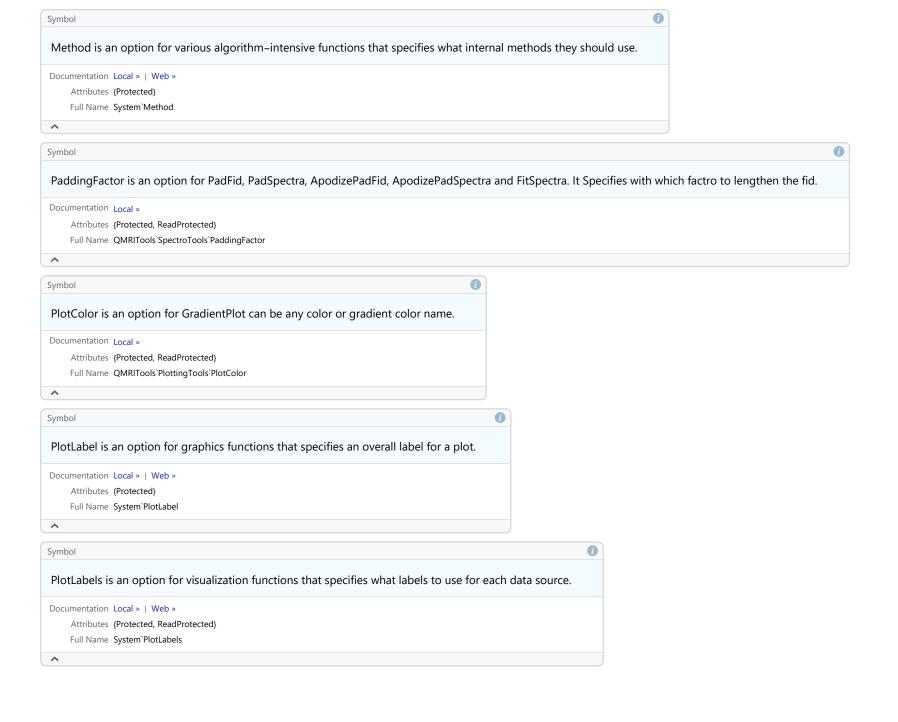
Documentation Local »

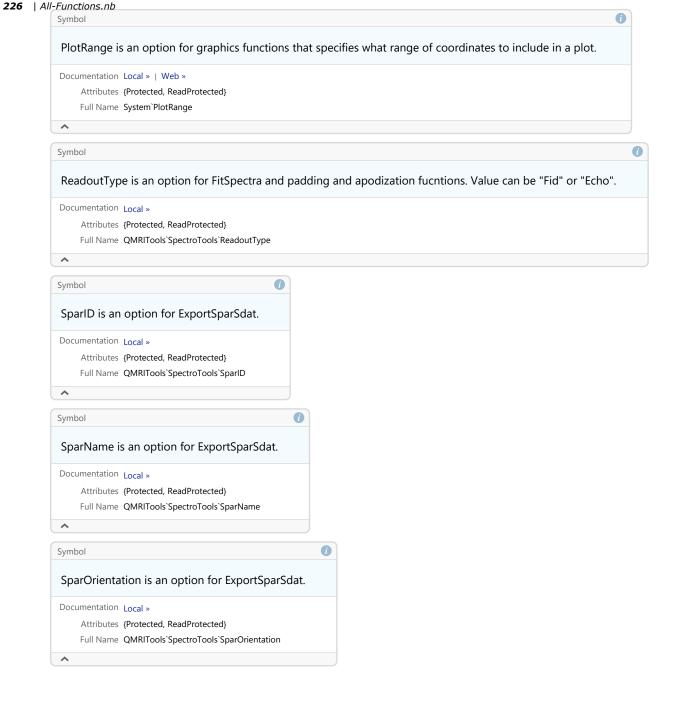
Attributes {Protected, ReadProtected}}

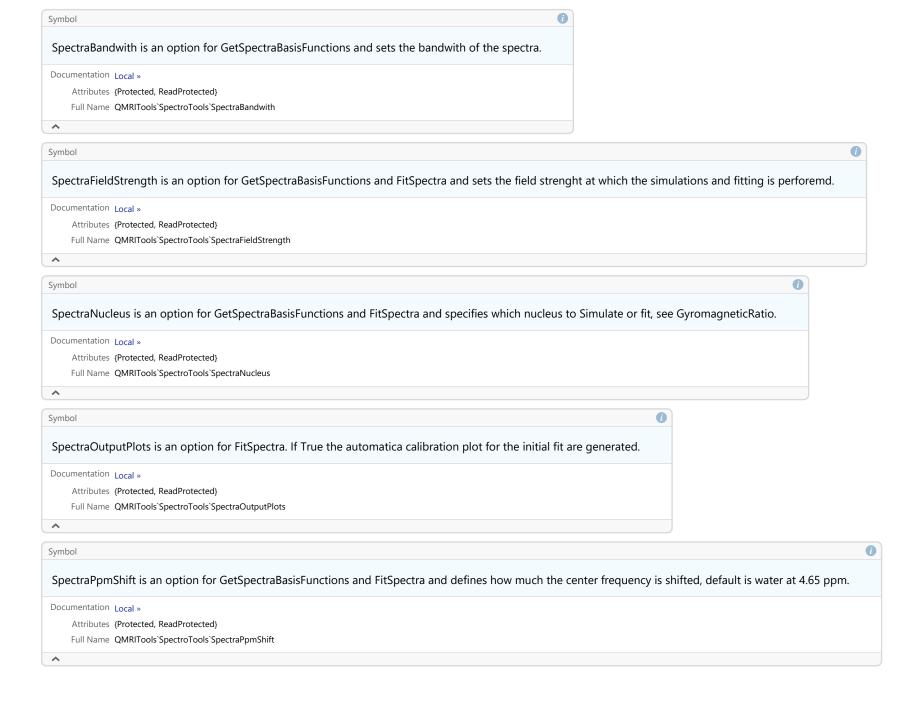
Full Name QMRITools`SpectroTools`ApodizationFunction

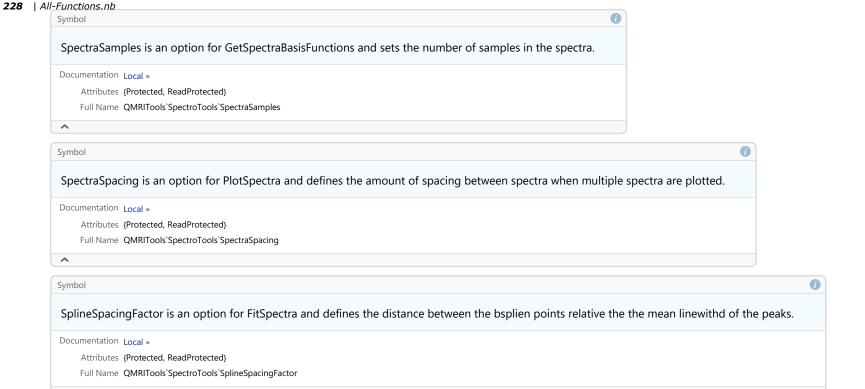






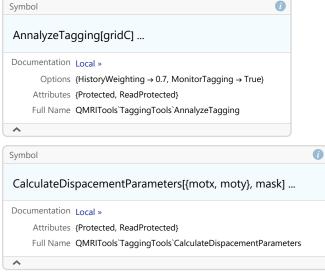




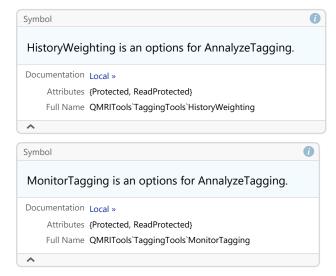


TaggingTools

Functions

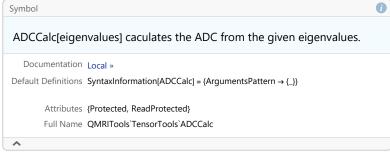


Options



TensorTools

Functions



```
AngleCalc[data, vector] calculates the angel between the vector and the data. Data shoud be an array of dimensions {xxxx,3}.

Documentation Local »

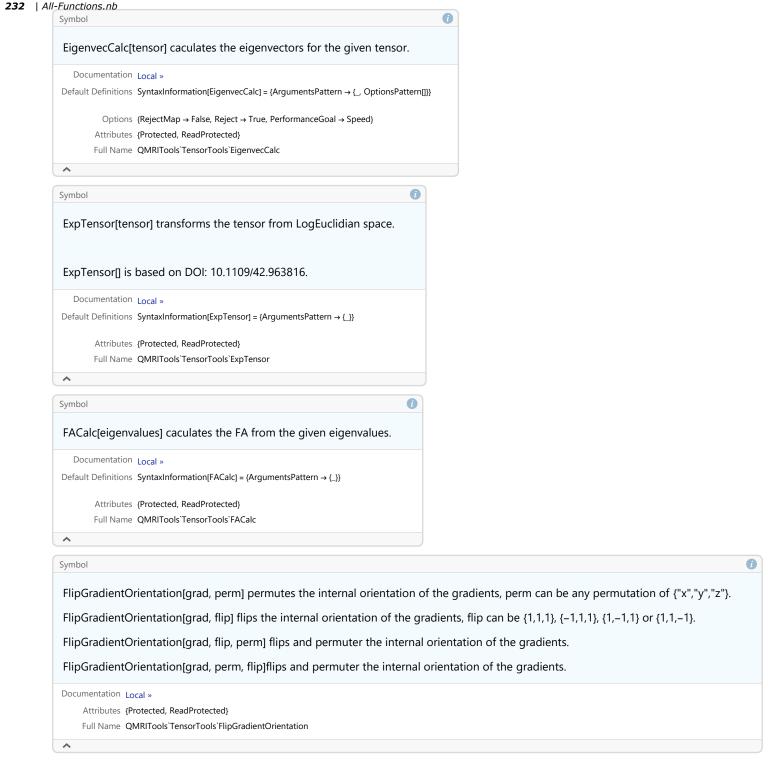
Default Definitions SyntaxInformation[AngleCalc] = {ArgumentsPattern → {..., OptionsPattern[]}}

Options Distribution → 0-180

Attributes {Protected, ReadProtected}

Full Name QMRITools'TensorTools'AngleCalc
```

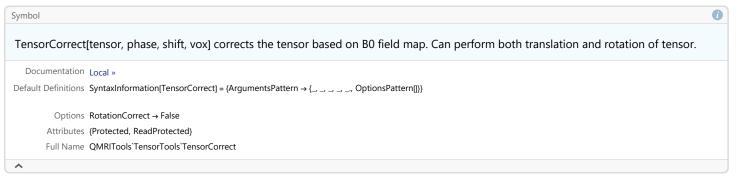
```
Symbol
DriftCorrect[data, bval] dirft corrects the data using the signals of the lowest bvalue that has 6 or more unique volumes.
For the function to work optimal it is best to have these volumes evenly spread througout thet data and for the first and last volume to have this low bvalue.
DriftCorrect[] is based on DOI: 10.1002/mrm.26124.
   Documentation Local »
Default Definitions SyntaxInformation[DriftCorrect] = {ArgumentsPattern → {_, _, _, _, OptionsPattern[]}}
          Options {NormalizeSignal → True, UseMask → True}
        Attributes {Protected, ReadProtected}
        Full Name QMRITools`TensorTools`DriftCorrect
^
Symbol
ECalc[eigenvalues] caculates the E from the given eigenvalues.
   Documentation Local »
Default Definitions SyntaxInformation[ECalc] = {ArgumentsPattern → {_, OptionsPattern[]}}
          Options MonitorCalc → True
        Attributes {Protected, ReadProtected}
        Full Name QMRITools`TensorTools`ECalc
\wedge
Symbol
EigensysCalc[tensor] caculates the eigensystem for the given tensor.
   Documentation Local »
\label{eq:definitions} Default\ Definitions\ \ SyntaxInformation[EigensysCalc] = \{ArgumentsPattern \rightarrow \{\_,\ OptionsPattern[]\}\}
          Options {RejectMap → False, Reject → True, PerformanceGoal → Speed}
        Attributes {Protected, ReadProtected}
        Full Name QMRITools`TensorTools`EigensysCalc
\wedge
Symbol
EigenvalCalc[tensor] caculates the eigenvalues for the given tensor.
   Documentation Local »
\label{eq:definitions} Default \ Definitions \ \ SyntaxInformation[EigenvalCalc] = \{ArgumentsPattern \rightarrow \{\_, \ OptionsPattern[]\}\}
          Options {RejectMap → False, Reject → True, PerformanceGoal → Speed}
        Attributes {Protected, ReadProtected}
        Full Name QMRITools`TensorTools`EigenvalCalc
```



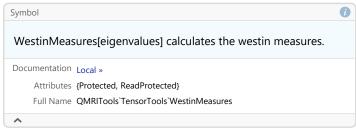
```
0
Symbol
FlipTensorOrientation[tens, perm] permutes the internal orientation of the tensor, perm can be any permutation of {"x","y","z"}.
FlipTensorOrientation[tens, flip] flips the internal orientation of the tensor, flip can be {1,1,1}, {-1,1,1}, {1,-1,1} or {1,1,-1}.
FlipTensorOrientation[tens, flip, perm] flips and permuter the internal orientation of the tensor.
FlipTensorOrientation[tens, perm, flip]flips and permuter the internal orientation of the tensor.
  Documentation Local »
Default Definitions SyntaxInformation[FlipTensorOrientation] = {ArgumentsPattern → {_, _, _,}}
       Attributes (Protected, ReadProtected)
       Full Name QMRITools`TensorTools`FlipTensorOrientation
^
                                                                            0
Symbol
LogTensor[tensor] transforms the tensor to LogEuclidian space.
LogTensor[] is based on DOI: 10.1109/42.963816.
  Documentation Local »
Default Definitions SyntaxInformation[LogTensor] = {ArgumentsPattern → {_}}
       Attributes {Protected, ReadProtected}
       Full Name QMRITools`TensorTools`LogTensor
\wedge
Symbol
ParameterCalc[tensor] caculates the eigenvalues and MD and FA from the given tensor. The parameters are I1, I2, I3, MD and FA. I1, I2, I3, MD are in (10^-3 mm^2/s).
  Documentation Local »
Default Definitions SyntaxInformation[ParameterCalc] = {ArgumentsPattern → {_, OptionsPattern[]}}
         Options {Reject → False, PerformanceGoal → Speed}
       Attributes {Protected, ReadProtected}
       Full Name QMRITools`TensorTools`ParameterCalc
\wedge
                                                                                                                                                               6
Symbol
RemovelsoImages[data, grad, bval] Romoves the ISO images from the philips scanner from the data. ISO images have g={0,0,0} and b>0.
  Documentation Local »
Default\ Definitions\ \ SyntaxInformation[RemovelsoImages] = \{ArgumentsPattern \rightarrow \{\_,\_,\_\}\}
       Attributes (Protected, ReadProtected)
       Full Name QMRITools'TensorTools'RemovelsoImages
\wedge
```

```
234 | All-Functions.nb
            Symbol
            ResidualCalc[dti,{tensor,s0},gradients,bvector] calculates the tensor residuals for the given dataset.
            ResidualCalc[dti,{tensor,s0},outlier,gradients,bvector] calculates the tensor residuals for the given dataset taking in account the outliers.
             ResidualCalc[dti,{tensor,s0},bmat] calculates the tensor residuals for the given dataset.
             ResidualCalc[dti,{tensor,s0},outlier,bmat] calculates the tensor residuals for the given dataset taking in account the outliers.
            ResidualCalc[dti,tensor,gradients,bvector] calculates the tensor residuals for the given dataset. Tensor must contain Log[s0].
             ResidualCalc[dti,tensor,outlier,gradients,bvector] calculates the tensor residuals for the given dataset taking in account the outliers. Tensor must contain Log[s0].
            ResidualCalc[dti,tensor,bmat] calculates the tensor residuals for the given dataset. Tensor must contain Log[s0].
             ResidualCalc[dti,tensor,outlier,bmat] calculates the tensor residuals for the given dataset taking in account the outliers. Tensor must contain Log[s0].
              Documentation Local »
            Default Definitions SyntaxInformation[ResidualCalc] = {ArgumentsPattern → {__, _, _, _, OptionsPattern[]}}
                     Options MeanRes → All
                   Attributes {Protected, ReadProtected}
                   Full Name OMRITools'TensorTools'ResidualCalc
            ^
                                                                                                                                                    0
            Symbol
            SigmaCalc[dti,grad,bvec] calculates the noise sigma based on the tensor residual, using a blur factor of 10.
            SigmaCalc[dti,tens,grad,bvec] calculates the noise sigma based on the tensor residual, using a blur factor of 10.
            SigmaCalc[dti,grad,bvec,blur] calculates the noise sigma based on the tensor residual, If blur is 1 ther is no blurring.
            SigmaCalc[dti,tens,grad,bvec,blur] calculates the noise sigma based on the tensor residual. If blur is 1 ther is no blurring.
              Documentation Local »
            Default Definitions SyntaxInformation[SigmaCalc] = {ArgumentsPattern → {_, _, _, _, _, _, OptionsPattern[]}}
                     Options FilterShape → Median
                   Attributes {Protected, ReadProtected}
                   Full Name QMRITools'TensorTools'SigmaCalc
            \wedge
            Symbol
            SortDiffusionData[data, grad, bval] sorts the diffusion datasets grad and bval for magnitude of bvalue.
              Documentation Local »
            Default Definitions SyntaxInformation[SortDiffusionData] = {ArgumentsPattern → {_, _, _}}
                   Attributes {Protected, ReadProtected}
                   Full Name QMRITools`TensorTools`SortDiffusionData
```

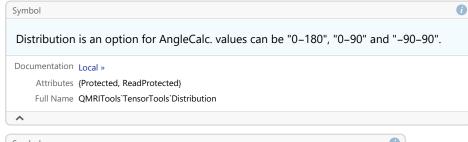
Symbol TensorCalc[data, gradients, bvalue] calculates the diffusion tensor for the given dataset. Allows for one unweighted image and one b value. Gradient directions must be in the form {{x1,y1,z1}, ..., {xn,yn,zn}} without the unweighted gradient direction. bvalue is a singe number indicating the b-value used. TensorCalc[data, gradients, bvec] calculates the diffusion tensor for the given dataset. allows for multiple unweighted images and multiple bvalues. allows for differnt tensor fitting methods. gradient directions must be in the form {{x1,y1,z1}, ..., {xn,yn,zn}} with the unweighted direction as {0,0,0}. bvec the bvector, with a bvalue defined for each gradient direction. b value for unweighted images is 0. TensorCalc[data, bmatix] calculates the diffusion tensor for the given dataset. allows for multiple unweighted images and multiple bvalues. bmat is the bmatrix which can be generated usiong Bmatrix. The bvalue assumed to be is in s/mm² and therfore the output is in mm²/2. TensorCalc[] is based on DOI: 10.1016/j.neuroimage.2013.05.028 and 10.1002/mrm.25165. Documentation Local » Default Definitions SyntaxInformation[TensorCalc] = {ArgumentsPattern → {_, _, _, OptionsPattern[]}} Options > MonitorCalc → True ... (6 total) Attributes {Protected, ReadProtected} Full Name QMRITools`TensorTools`TensorCalc \wedge

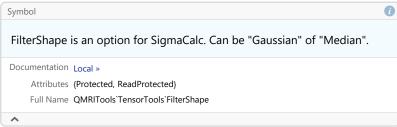


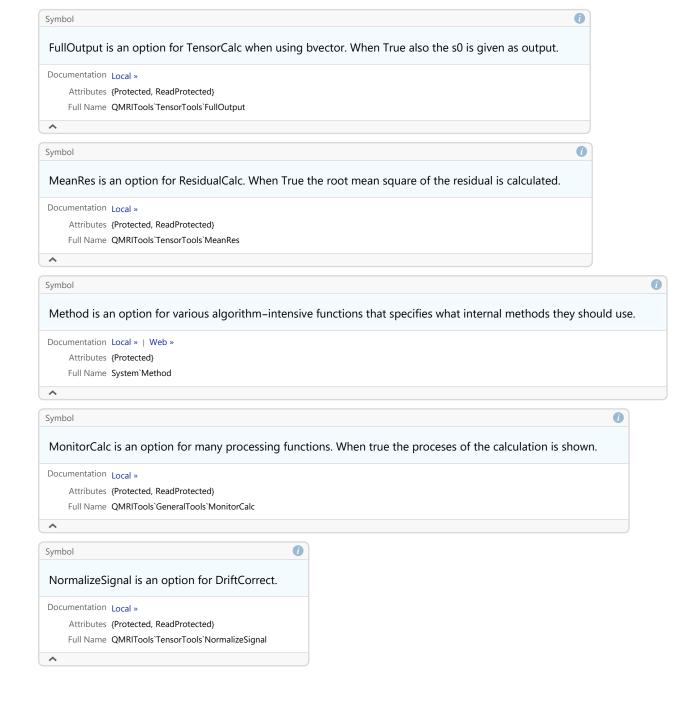


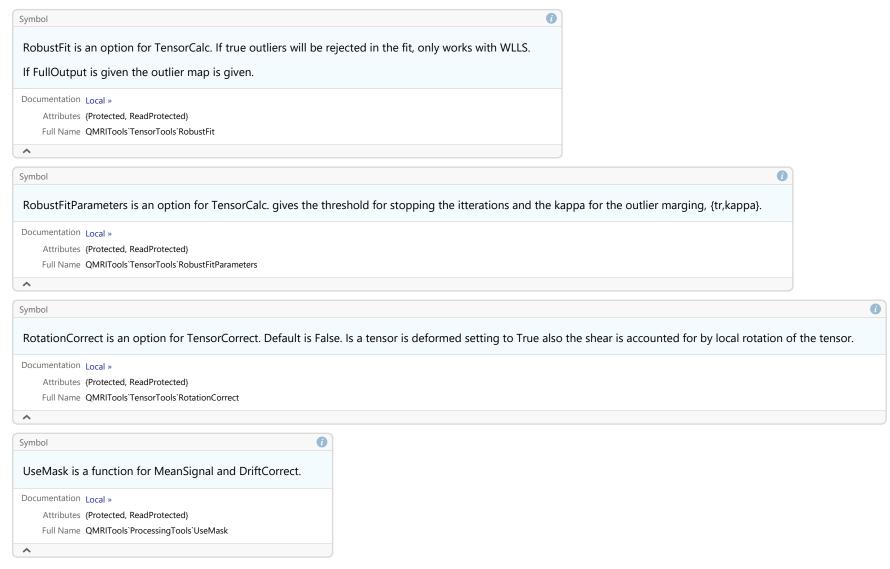


Options



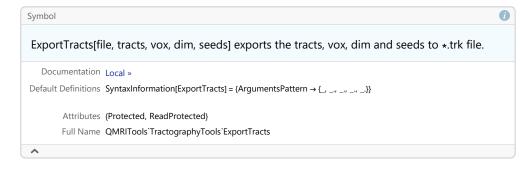


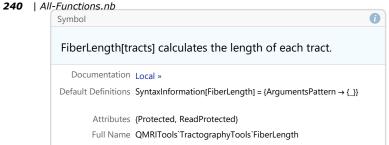


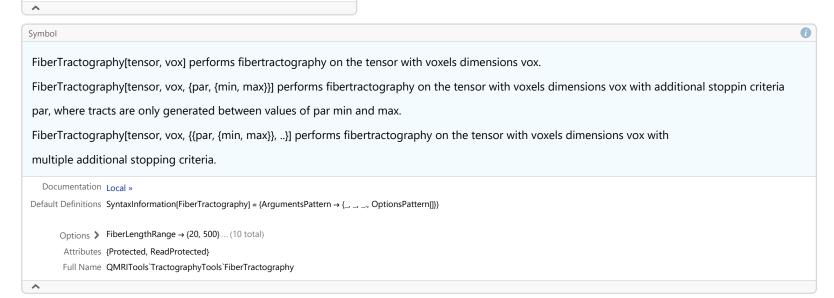


TractographyTools

Functions







0

```
Symbol
FilterTracts[tracts, vox, {select...}] filters the tracts based on the list of select criteria.
Select criteria are defined as {"logic",{"how", criteria}}.
The "logic" parameter can be "and", "or" and "not".
The "how" parameter can be:
         - "x", "y", or "z" for slice selection, here criteria is a slice number
         - "thourgh" for selecting tract that go through a roi, here criteria is a 3D mask.
         - "within" for selecting tract that fit fully within the roi, here criteria is a 3D mask.
         - "partwithin" for selecting the part of the tracts that fall within the roi, here criteria is a 3D mask.
Any number of select criteria can be listed.
  Documentation Local »
Default Definitions SyntaxInformation[FilterTracts] = {ArgumentsPattern → {_, _, _, _, OptionsPattern[]}}
         Options FiberLengthRange → {20, 500}
       Attributes {Protected, ReadProtected}
      Full Name QMRITools`TractographyTools`FilterTracts
^
```

Symbol

FindTensorPermutation[tensor, vox] performs tractography for all tensor permutations and gives back the one that has the longest tracts.

FindTensorPermutation[tensor, vox, {par, {min, max}}] same but with additional stoppin criteria par, where tracts are only generated between values of par min and max.

FindTensorPermutation[tensor, vox, {{par, {min, max}}, ...}] same but with multiple additional stopping criteria.

Ouput = {permutations, flips, plot}

FindTensorPermutation[] is based on DOI: 10.1016/j.media.2014.05.012.

Documentation Local »

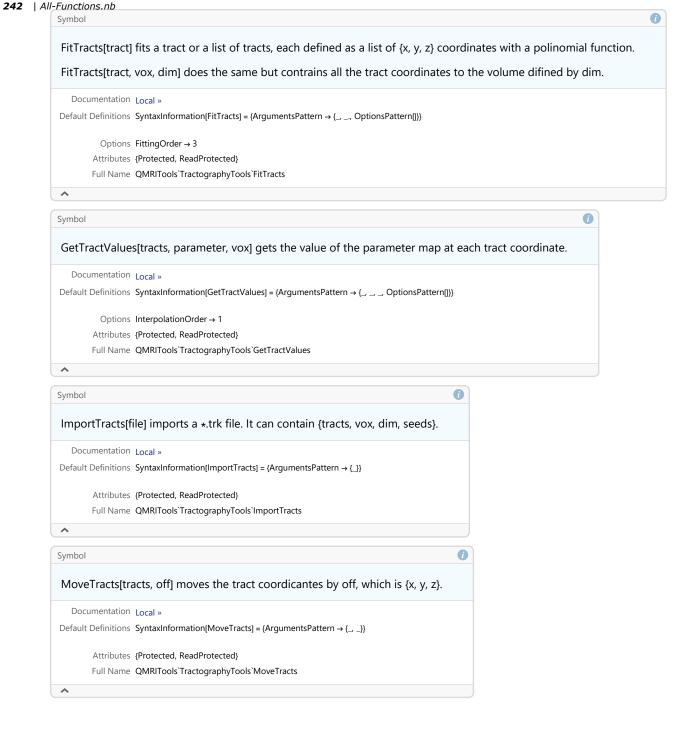
Default Definitions SyntaxInformation[FindTensorPermutation] = {ArgumentsPattern → {_, _, _, _, OptionsPattern[]}}

Options \rightarrow FiberLengthRange \rightarrow {20, 500}...(7 total)

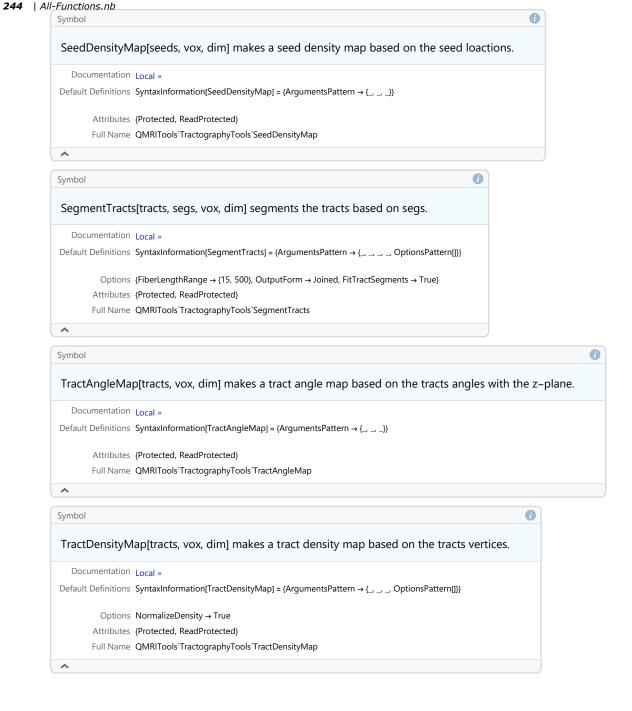
Attributes {Protected, ReadProtected}

Full Name QMRITools`TractographyTools`FindTensorPermutation

^



```
Symbol
PlotSegmentedTracts[tracts, segments, dim, vox] plots the tracts after segmenting each segments.
PlotSegmentedTracts[tracts, segments, bones, dim, vox] plots the tracts after segmenting each segments also rendering a bone volume.
   Documentation Local »
Default Definitions SyntaxInformation[PlotSegmentedTracts] = {ArgumentsPattern → {_ _ _ _ _ _ _ _ , OptionsPattern[]}}
       Options ➤ MaxTracts → 5000 ... (6 total)
        Attributes {Protected, ReadProtected}
       Full Name QMRITools`TractographyTools`PlotSegmentedTracts
                                                                                                                                                 0
Symbol
PlotTracts[tracts, vox] plots the tracts assuming an Boxratio based on vox.
PlotTracts[tracts, vox, dim] plots the tracts assuming an Boxratio based on vox with a PlotRange spanning the full dim.
   Documentation Local »
Default\ Definitions\ \ SyntaxInformation[PlotTracts] = \{ArgumentsPattern \rightarrow \{\_, \_, \_, OptionsPattern[]\}\}
       Options ➤ MaxTracts → 2000 ... (10 total)
        Attributes {Protected, ReadProtected}
       Full Name QMRITools`TractographyTools`PlotTracts
^
Symbol
ResampleTracts[tracts, n] resample each Tract to exactly n vertices.
  Documentation Local »
Default Definitions SyntaxInformation[ResampleTracts] = {ArgumentsPattern \rightarrow {_, _}}
        Attributes {Protected, ReadProtected}
        Full Name QMRITools`TractographyTools`ResampleTracts
\wedge
Symbol
RescaleTracts[tracts, sc] scales the tract coordinates by 1/sc, which is {x, y, z} or single number.
   Documentation Local »
Default\ Definitions\ \ SyntaxInformation[RescaleTracts] = \{ArgumentsPattern \rightarrow \{\_,\_\}\}
        Attributes {Protected, ReadProtected}
        Full Name QMRITools`TractographyTools`RescaleTracts
\wedge
```



```
Symbol
TractLengthMap[tracts, vox, dim] makes a tract length map based on the tracts lengths.
  Documentation Local »
 Default \ Definitions \ \ SyntaxInformation[TractLengthMap] = \{ArgumentsPattern \rightarrow \{\_,\_,\_\}\} 
        Attributes {Protected, ReadProtected}
       Full Name QMRITools`TractographyTools`TractLengthMap
```

