Defined packages and functions to be loaded are:

- QMRITools`LoggingTools` with functions:

{AddToLog, CheckFile, DirectoryTree, ExportDirectoryTree, ExportLog, ImportLog, MakeCheckFile, PrintDirectoryTree, ResetLog, ShowLog}

- OMRITools`GeneralTools` with functions:

{ApplyCrop, AutoCropData, BSplineCurveFit, CenterVoxel, ClearTemporaryVariables, CompilebleFunctions, ConvertExtension, CropAlways, CropData, CropInit, CropOutput, CropPadding, CutData, DataToVector, DecomposeAffineMatrix, DecomposeScaleMatrix, DevideNoZero, DynamicPartition, EmptyDirectoryQ, ExpNoZero, ExtractDemoData, FileSelect, FindCrop, FindMaxDimensions, GetAssetLocation, GridData, GridData3D, GyromagneticRatio, LapFilter, LLeastSquares, LogNoZero, MADNoZero, MakeCoordinates, MakeIntFunction, MeanNoZero, MedFilter, MedianNoZero, MemoryUsage, MonitorCalc, NiiFileExistQ, NNLeastSquares, OutputWeights, PadDirection, PadToDimensions, PadValue, QMRIToolsFuncPrint, QMRIToolsFunctions, QMRIToolsPackages, QuaternionToRotationMatrix, QuaternionVectorToRotationMatrix, RescaleData, ReverseCrop, ReverseDimensions, RMSNoZero, RotateDimensionsLeft, RotateDimensionsRight, RotationMatrixToQuaternion, RotationMatrixToQuaternionVector, SaveImage, SignNoZero, SplineKnotsNumber, SplineRegularization, Squeeze, StdFilter, StichData, StringPadInteger, SumOfSquares, TensMat, TensVec, VectorToData}

- QMRITools`MaskingTools` with functions:

{DilateMask, GetMaskData, GetMaskOnly, GetMaskOutput, HomoginizeData, ImportITKLabels, Mask, MaskClosing, MaskComponents,

MaskData, MaskDilation, MaskFiltKernel, MaskSmoothing, MeanSignal, MergeSegmentations, NormalizeData, NormalizeMeanData, NormalizeMethod,

RemoveMaskOverlaps, RescaleSegmentation, ROIMask, SegmentMask, SmoothItterations, SmoothMask, SmoothSegmentation, SplitSegmentations, UseMask}

- QMRITools`NiftiTools` with functions:

{CompressNii, CompressNiiFiles, CorrectNiiOrientation, DcmToNii, DeleteOutputFolder, ExportBmat, ExportBval, ExportBvac, ExportNii, ExtractNiiFiles, FlipBvec, GetNiiOrientation, ImportBmat, ImportBval, ImportBvalvec, ImportBvec, ImportExploreDTItens, ImportNii, ImportNiiDiff, ImportNiiDix, ImportNiiT1, ImportNiiT2, MakeNiiOrentationQ, MakeNiiOrentationS, NiiDataType, NiiLegacy, NiiMethod, NiiOffset, NiiScaling, NiiSliceCode, RotateGradients, UseSubfolders, UseVersion}

- OMRITools`ElastixTools` with functions:

{AffineDirections, BsplineDirections, BsplineSpacing, DeleteTempDirectory, FindTransform, HistogramBins, HistogramBinsA,
InterpolationOrderReg, InterpolationOrderRegA, Iterations, IterationsA, MethodReg, MethodRegA, NumberSamples, NumberSamplesA,
OutputImage, OutputTransformation, PCAComponents, PrintTempDirectory, ReadTransformParameters, RegisterCardiacData, RegisterData,
RegisterDataSplit, RegisterDataTransform, RegisterDataTransformSplit, RegisterDiffusionData, RegisterDiffusionDataSplit, RegisterTensorData,
RegistrationTarget, Resolutions, ResolutionsA, ShowMetric, SplitMethod, TempDirectory, TransformData, UseGPU, \$debugElastix}

- QMRITools`PlottingTools` with functions:

{ColorFAPlot, ContourColor, ContourColorRange, ContourOpacity, ContourQuality, ContourSize, DropSlices, GetSliceData, GetSlicePositions,
GradientPlot, ImageLegend, ImageOrientation, ListSpherePlot, MakeCheckPlot, MakeSliceImages, NormalizeIVIM, PeakNumber, PlotColor, PlotContour,
PlotCorrection, PlotData, PlotData3D, PlotDefGrid, PlotDuty, PlotIVIM, PlotMoments, PlotSequence, PlotSpace, PositiveZ, SphereColor, SphereSize}

- OMRITools`MuscleBidsTools` with functions:

{AddToJson, BidsDcmToNii, BidsIncludeSession, CheckDataDiscription, DeleteAfterConversion, ExtractFromJSON, GenerateBidsFileName,
GenerateBidsFolderName, GenerateBidsName, GetConfig, GetJSONPosition, ImportJSON, MergeJSON, MuscleBidsConvert, MuscleBidsMerge, MuscleBidsProcess,
PartitionBidsFolderName, PartitionBidsName, SelectBidsFolders, SelectBidsSessions, SelectBidsSubjects, SelectSubjects, VersionCheck, ViewConfig}

2 | All-Functions.nb - QMRITools`DixonTools` with functions:

{DixonAmplitudes, DixonBipolar, DixonClipFraction, DixonComplexOutput, DixonCorrectT1, DixonFieldStrength, DixonFilterInput, DixonFilterOutput, DixonFilterSize, DixonFilterType, DixonFrequencies, DixonIterations, DixonMaskThreshhold, DixonNucleus, DixonPhase, DixonPrecessions, DixonReconstruct, DixonTollerance, DixonToPercent, FindInPhaseEchos, FixDixonFlips, MonitorUnwrap, OptimizeDixonEcho, SimulateDixonSignal, Unwrap, UnwrapDCT, UnwrapDimension, UnwrapList, UnwrapSplit, UnwrapThresh, Wrap}

- QMRITools`IVIMTools` with functions:

{BayesianIVIMFit2, BayesianIVIMFit3, ChainSteps, CorrectPar, CorrectParMap, FConvert, FConverti, FilterMaps, FilterSize, FilterType, FitConstrains, FixPseudoDiff5, FixPseudoDiff5D, FracCorrect, HistogramPar, IVIMCalc, IVIMComponents, IVIMConstrained, IVIMConstrains, IVIMCorrectData, IVIMFixed, IVIMFunction, IVIMResiduals, IVIMTensFit, MeanBvalueSignal, MonitorIVIMCalc, OutputSamples, ThetaConvi, UpdateStep}

- QMRITools`DenoiseTools` with functions:

{AnisoFilterData, AnisoFilterSteps, AnisoFilterTensor, AnisoItterations, AnisoKappa, AnisoKernel, AnisoStepTime,

- OMRITools`CardiacTools` with functions:

{AxesMethod, BackgroundValue, BloodMaskRange, BullPlotMethod, BullseyePlot, CalculateWallMap, CardiacCoordinateSystem, CardiacSegment, CardiacSegmentGUI, CentralAxes, CreateHeart, CutOffMethod, DistanceMeasure, DropSamples, ECVCalc, ExcludeSlices, GetMaskSegmentPoints, GetSegmentLines, GetSegmentSlices, GridLineSpacing, GroupPerSegment, HelixAngleCalc, LCMMethod, LinesToSegmentIndex, MakeECVBloodMask, MakeLineImage, MakeMaskImage, MakeSegmentPlots, MaskHelix, MaskToLines, MaskWallMap, OutputCheckImage, PlotSegmentMask, PlotSegments, RadialSamples, ReverseDirection, ReversePoints, RowSize, SegmentationMethod, SegmentLinesToMask, SegmentsPerSlice, ShowOutliers, ShowPlot, SmoothHelix, StartPoints, StartSlices, TextNumberForm, TextOffset, TextSize, TransmuralPlot}

- QMRITools`RelaxometryTools` with functions:

{CalibrateEPGT2Fit, CreateT2Dictionary, DictB1Range, DictionaryMinSearch, DictT2fRange, DictT2fValue, DictT2IncludeWater,
DictT2Range, EPGCalibrate, EPGFatShift, EPGFitFat, EPGFitPoints, EPGMethod, EPGMethodCal, EPGRelaxPars, EPGSignal, EPGSmoothB1, EPGT2Fit,
NonLinearEPGFit, OutputCalibration, ShiftPulseProfile, T1Fit, T1rhoFit, T2Fit, TriExponentialT2Fit, WaterFatShift, WaterFatShiftDirection}

AnisoWeightType, DeNoise, DenoiseCSIdata, DenoiseDynamicSpectraData, DeNoiseIterations, DeNoiseKernel, DeNoiseMonitor, NNDeNoise, NNThreshhold, PCAClipping, PCADeNoise, PCADeNoiseFit, PCAKernel, PCANoiseSigma, PCAOutput, PCATollerance, PCAWeighting, WeightMapCalc)

- QMRITools`GradientTools` with functions:

{Bmatrix, BmatrixCalc, BmatrixConv, BmatrixInv, BmatrixRot, BmatrixToggle, CalculateMoments, ConditionCalc, ConditionNumberCalc, ConvertGrads, CorrectBmatrix, CorrectGradients, EnergyCalc, FinalGrads, FindOrder, FlipAxes, FlipGrad, FullGrad, FullSphere, GenerateGradients, GenerateGradientsGUI, GetGradientScanOrder, GetSliceNormal, GetSliceNormalDir, GradBmatrix, GradSeq, GradType, ImportGradObj, OrderSpan, OutputPlot, OutputType, OverPlusCalc, PhaseEncoding, Runs, SelectBvalueData, Steps, StepSizeI, SwitchAxes, UniqueBvalPosition, UnitMulti, UseGrad, VisualOpt}

- OMRITools`TensorTools` with functions:

{ADCCalc, AngleCalc, AngleMap, ConcatenateDiffusionData, Correct, Deriv, Distribution, DriftCorrect, ECalc, EigensysCalc, EigenvalCalc, EigenvecCalc, ExpTensor, FACalc, FilterShape, FlipGradientOrientation, FlipTensorOrientation, FullOutput, LogTensor, MeanRes, NormalizeSignal, ParameterCalc, Reject, RejectMap, RemoveIsoImages, ResidualCalc, RobustFit, RobustFitParameters, RotationCorrect, SigmaCalc, SortDiffusionData, TensorCalc, TensorCorrect, TransformTensor, WestinMeasures}

- OMRITools`JcouplingTools` with functions:

{CenterFrequency, FieldStrength, GetSpinSystem, Linewidth, LinewidthShape, MakeSpinSystem, ReadoutBandwith,
ReadoutMethod, ReadoutOutput, ReadoutPhase, ReadoutSamples, SequencePulseAcquire, SequenceSpaceEcho, SequenceSpinEcho,
SequenceSteam, SequenceTSE, SimAddPhase, SimEvolve, SimHamiltonian, SimNucleus, SimReadout, SimRotate, SimSignal, SimSpoil, SysTable}

- OMRITools`SpectroTools` with functions:

{ApodizationFunction, ApodizeEcho, ApodizeFid, ApodizePadEcho, ApodizePadFid, ApodizePadSpectra, ApodizeSpectra, BasisSequence, ChangeDwellTimeFid, CompareFidFitPlot, CompareSpectraFitPlot, CorrectTESpec, CSIInterface, ExportSparSdat, FindSpectraPpmShift, FineTuneFit, FitLineShape, FitSpectra, FitSpectraResultTable, GetGyro, GetPpmRange, GetSpectraBasisFunctions, GetTimePpmRange, GetTimeRange, ImportSparSdat, InitializeFit, MakeSpectraResultPlot, PaddingFactor, PadEcho, PadFid, PadSpectra, PhaseCorrectSpectra, PhaseShiftSpectra, PlotCSIData, PlotFid, PlotSpectra, ReadjMRUI, ReadoutType, ShiftSpectra, SparID, SparName, SparOrientation, SpectraBandwith, SpectraFieldStrength, SpectraFitResult, SpectraNucleus, SpectraOutputPlots, SpectraPpmShift, SpectraSamples, SpectraSpacing, SplineSpacingFactor, TimeShiftEcho, TimeShiftFid, TimeShiftFidV}

- OMRITools`ReconstructionTools` with functions:

{AcquisitionMethod, CoilCombine, CoilSamples, CoilWeightedRecon, CoilWeightedReconCSI, DeconvolutionMethod, DeconvolveCSIdata, EchoShiftData, FourierKspace2D, FourierKspace3D, FourierKspaceCSI, FourierRescaleData, FourierShift, FourierShifted, HammingFilter, HammingFilterCSI, HammingFilterData, InverseFourierShift, InverseFourierShifted, MakeHammingFilter, MakeSense, MeanType, NoiseCorrelation, NoiseCovariance, NormalizeOutputSpectra, NormalizeSpectra, OrderKspace, OutputSense, ReadListData, ReconFilter, RescaleRecon, SagitalTranspose, SenseRescale, ShiftedFourier, ShiftedInverseFourier, TotalType, WienerRegularization}

- QMRITools`TractographyTools` with functions:

{CombineROIs, FiberAngle, FiberLength, FiberLengthRange, FiberTractography, FilterTracts, FindTensorPermutation, FittingOrder, FitTract, GetTractValues, MaxSeedPoints, MaxTracts, NormalizeDensity, PartTracts, PlotTracts, SeedDensityMap, SelectTractInVol, SelectTractPartInVol, SelectTracts, SelectTractTroughPlane, SelectTractTroughVol, StepSize, StopThreshhold, TensorFilps, TensorPermutations, TracMonitor, TractAngleMap, TractColoring, TractDensityMap, TractLengthMap}

- QMRITools`VisteTools` with functions:

{BinaryType, DatRead, DatWrite, DTItoolExp, DTItoolExpFile, DTItoolExpInd, DTItoolExpTens, ExportVol, ImportDTI, ImportVol, LoadFiberTracts}

- QMRITools`ProcessingTools` with functions:

{B1EqualPower, B1FilterData, B1MapCalc, B1Masking, B1MaxPower, B1Output, B1Scaling, B1ShimMethod, B1Shimming, ColorValue, CombineB1, CorrectJoinSetMotion, DataTransformation, DatTot, DatTotXLS, ErrorPlot, FiberDensityMap, FiberLengths, FindOutliers, FitData, FitFunction, FitOutput, GetMaskMeans, Hist, Hist2, InvertDataset, JoinSets, JoinSetSplit, MaskCompartment, MeanMethod, MeanRange, MeanStd, MedCouple, MotionCorrectSets, NormalizeOverlap, NormalizeSets, NumberTableForm, OutlierIncludeZero, OutlierIterations, OutlierMethod, OutlierOutput, OutlierRange, OutputSNR, PaddOverlap, ParameterFit, ParameterFit2, ReferenceB1, ReverseData, ReverseSets, RotateData, RotateTensor, Scaling, SeedDensity, SetupDataStructure, SmartMask, SmartMaskOutput, SmartMethod, SmoothSNR, SNRCalc, SNRMapCalc, SplitSets, Strictness, TableMethod}

- OMRITools`FasciculationTools` with functions:

{ActivationBackground, ActivationIterations, ActivationOutput, ActivationSize, ActivationThreshold, AnalyzeActivations, EvaluateActivation, FindActivations, IgnoreSlices, SelectActivations, ThresholdMethod}

– QMRITools`SimulationTools` with functions:

{AddNoise, BlochSeries, CalculateGfactor, CreateDiffData, ErnstAngle, FatFieldStrength, GESignal, GetPulseProfile, GfactorSimulation, GOutput, GRegularization, MagnetizationVector, NoiseSize, NoiseType, PlotSimulation, PlotSimulationAngle, PlotSimulationAngleHist, PlotSimulationHist, PlotSimulationVec, Pulses, ReportFits, Signal, SimAngleParameters, SimParameters, SimulateDualTR, SimulateSliceEPG, SliceRange, SliceRangeSamples, SortVecs, Tensor, TensOutput}

- QMRITools`PhysiologyTools` with functions:

{AlignRespLog, ImportPhyslog, ImportRespirect, OutputMethod, PlotPhyslog, PlotRespiract, SampleStep}

- OMRITools`CoilTools` with functions:

{CoilArrayPlot, CoilSNRCalc, CoilSurfaceVoxelSize, FindCoilPosition, LoadCoilSetup, LoadCoilTarget, MakeCoilLayout, MakeNoisePlots, MakeWeightMask, OutputCoilSurface}

– QMRITools`TaggingTools` with functions:

{AnnalyzeTagging, CalculateDispacementParameters, HistoryWeighting, MonitorTagging}

- QMRITools`ImportTools` with functions:

```
4 | All-Functions.nb
       {BmatrixOut, BvalRead, ConvertDcm, GradRead, ReadBrukerDiff, ReadBvalue, ReadDicom,
        ReadDicomDiff, ReadDicomDir, ReadDicomDirDiff, ReadGradients, ReadVoxSize, RotateGradient, ScaleCorrect, ShiftPar
       Removing all local and global definitions of:
          - QMRITools`LoggingTools`
          - QMRITools`GeneralTools`
          - QMRITools`MaskingTools`
          - QMRITools`NiftiTools`
          - QMRITools`ElastixTools`
          QMRITools`PlottingTools`
          - QMRITools`MuscleBidsTools`
          - QMRITools`DixonTools`
          - QMRITools`IVIMTools`
          QMRITools`DenoiseTools`
          - QMRITools`CardiacTools`
          - QMRITools`RelaxometryTools`
          QMRITools`GradientTools`
          - QMRITools`TensorTools`
          – QMRITools`JcouplingTools`
          QMRITools`SpectroTools`
          QMRITools`ReconstructionTools`
          QMRITools`TractographyTools`
          - QMRITools`VisteTools`
          QMRITools`ProcessingTools`
          QMRITools`FasciculationTools`
          – QMRITools`SimulationTools`
          - QMRITools`PhysiologyTools`
          QMRITools`CoilTools`
          QMRITools`TaggingTools`
          QMRITools`ImportTools`
       Loading and protecting all definitions of:
          QMRITools`LoggingTools`
```

- QMRITools`GeneralTools`
- QMRITools`MaskingTools`
- QMRITools`NiftiTools`
- QMRITools`ElastixTools`
- QMRITools`PlottingTools`
- QMRITools`MuscleBidsTools`
- QMRITools`DixonTools`
- QMRITools`IVIMTools`
- QMRITools`DenoiseTools`
- QMRITools`CardiacTools`
- QMRITools`RelaxometryTools`
- QMRITools`GradientTools`
- QMRITools`TensorTools`
- QMRITools`JcouplingTools`
- QMRITools`SpectroTools`
- QMRITools`ReconstructionTools`
- QMRITools`TractographyTools`
- QMRITools`VisteTools`
- QMRITools`ProcessingTools`
- QMRITools`FasciculationTools`
- QMRITools`SimulationTools`
- QMRITools`PhysiologyTools`
- QMRITools`CoilTools`
- QMRITools`TaggingTools`
- QMRITools`ImportTools`

### **6** | All-Functions.nb In[26]:= Column@QMRIToolsPackages[] Out[26]= CardiacTools CoilTools DenoiseTools DixonTools ElastixTools FasciculationTools GeneralTools GradientTools ImportTools IVIMTools JcouplingTools LoggingTools MaskingTools MuscleBidsTools NiftiTools PhysiologyTools PlottingTools ProcessingTools ReconstructionTools RelaxometryTools SimulationTools SpectroTools TaggingTools TensorTools TractographyTools VisteTools

#### QMRIToolsFunctions[80]

#### **Functions**

ADCCalc	DatTotXLS	FourierShifted	LogNoZero	PlotSequence	<i>AU-Functions.nb</i>   <b>7</b> Signal
AddNoise	DatWrite	FracCorrect	LogTensor	PlotSimulation	SignNoZero
AddToJson	DcmToNii	FullGrad	MADNoZero	PlotSimulationAngle	SimAddPhase
AddToLog	DecomposeAffineMatrix	GenerateBidsFileName	MakeCheckFile	PlotSimulationAngleHist	SimAngleParamet
AlignRespLog	DecomposeScaleMatrix	GenerateBidsFolderName	MakeCoilLayout	PlotSimulationHist	SimEvolve
AnalyzeActivations	DeconvolveCSIdata	GenerateBidsName	MakeCoordinates	PlotSimulationVec	SimHamiltonian
AngleCalc	DeNoise	GenerateGradients	MakeECVBloodMask	PlotSpectra	SimParameters
AngleMap	DenoiseCSIdata	GenerateGradientsGUI	MakeHammingFilter	PlotTracts	SimReadout
AnisoFilterData	DenoiseDynamicSpectraData	GESignal	MakeIntFunction	PrintDirectoryTree	SimRotate
AnisoFilterTensor	Deriv	GetAssetLocation	MakeLineImage	Pulses	SimSignal
AnnalyzeTagging	DevideNoZero	GetConfig	MakeMaskImage	OMRIToolsFuncPrint	SimSpoil
ApodizeEcho	DictionaryMinSearch	GetGradientScanOrder	MakeNiiOrentationQ	OMRIToolsFunctions	SimulateDixonS:
ApodizeFid	DilateMask	GetGyro	MakeNiiOrentationS	QMRIToolsPackages	SimulateDualTR
ApodizePadEcho	DirectoryTree	GetJSONPosition	MakeNoisePlots	OuaternionToRotationMatrix	SimulateSliceE
ApodizePadFid	DixonPhase	GetMaskData	MakeSense	OuaternionVectorToRotationMatrix	SmartMask
ApodizePadSpectra	DixonReconstruct	GetMaskMeans	MakeSliceImages	RadialSample	SmoothMask
ApodizeSpectra	DixonToPercent	GetMaskSegmentPoints	MakeSpectraResultPlot	ReadBrukerDiff	SmoothSegmentat
ApplyCrop	DriftCorrect	GetNiiOrientation	MakeSpinSystem	ReadBvalue	SNRCalc
AutoCropData	DTItoolExp	GetPpmRange	MakeWeightMask	ReadDicom	SNRMapCalc
B1MapCalc	DTItoolExpFile	GetPulseProfile	Mask	ReadDicomDiff	SortDiffusionDa
B1Shimming	DTItoolExpInd	GetSegmentLines	MaskData	ReadDicomDir	SpectraFitResul
BayesianIVIMFit2	DTItoolExpTens	GetSegmentSlices	MaskHelix	ReadDicomDirDiff	SplitSegmentati
-	•	GetSliceData	MaskToLines	ReadGradients	
BayesianIVIMFit3 BidsDcmToNii	DynamicPartition ECalc	GetSliceData GetSliceNormal	MeanBvalueSignal	ReadjMRUI	SplitSets
			· ·	3	Squeeze
BlochSeries	ECVCalc	GetSliceNormalDir	MeanNoZero	ReadListData	StdFilter
Bmatrix	EigensysCalc	GetSlicePositions	MeanRange	ReadTransformParameters	StichData
BmatrixCalc	EigenvalCalc	GetSpectraBasisFunctions	MeanSignal	ReadVoxSize	StringPadIntege
BmatrixConv	EigenvecCalc	GetSpinSystem	MeanStd	RegisterCardiacData	SumOfSquares
BmatrixInv	EmptyDirectoryQ	GetTimePpmRange	MeanType	RegisterData	SysTable
BmatrixRot	EnergyCalc	GetTimeRange	MedCouple	RegisterDataSplit	T1Fit
BmatrixToggle	EPGSignal	GetTractValues	MedFilter	RegisterDataTransform	T1rhoFit
BSplineCurveFit	EPGT2Fit	GfactorSimulation	MedianNoZero	RegisterDataTransformSplit	T2Fit
BullseyePlot	ErnstAngle -	GradBmatrix	MemoryUsage	RegisterDiffusionData	TensMat
BvalRead	ErrorPlot	GradientPlot	MergeJSON	RegisterDiffusionDataSplit	Tensor
CalculateDispacementParameters	EvaluateActivation	GradRead	MergeSegmentations	RegisterTensorData	TensorCalc
CalculateGfactor	ExcludeSlices	GradSeq	MonitorCalc	RemoveIsoImages	TensorCorrect
CalculateMoments	ExpNoZero	GridData	MuscleBidsConvert	RemoveMaskOverlaps	TensVec
CalculateWallMap	ExportBmat	GridData3D	MuscleBidsMerge	RescaleData	ThetaConv
CalibrateEPGT2Fit	ExportBval	GyromagneticRatio	MuscleBidsProcess	RescaleSegmentation	ThetaConvi
CardiacCoordinateSystem	ExportBvec	HammingFilterCSI	NiiFileExistQ	ResetLog	TimeShiftEcho
CardiacSegment	ExportDirectoryTree	HammingFilterData	NNDeNoise	ResidualCalc	TimeShiftFid
CardiacSegmentGUI	ExportLog	HelixAngleCalc	NNLeastSquares	ReverseCrop	TimeShiftFidV
CentralAxes	ExportNii	Hist	NoiseCorrelation	ReverseDimensions	TotalType
ChangeDwellTimeFid	ExportSparSdat	Hist2	NoiseCovariance	RMSNoZero	TractAngleMap
CheckDataDiscription	ExportVol	HistogramPar	NonLinearEPGFit	ROIMask	TractDensityMa
CheckFile	ExpTensor	HomoginizeData	NormalizeData	RotateData	TractLengthMap
ClearTemporaryVariables	ExtractDemoData	ImportBmat	NormalizeMeanData	RotateDimensionsLeft	TransformData
CoilCombine	ExtractFromJSON	ImportBval	NormalizeSpectra	RotateDimensionsRight	TransformTenso
CoilSNRCalc	ExtractNiiFiles	ImportBvalvec	NumberTableForm	RotateTensor	TransmuralPlot
CoilWeightedRecon	FACalc	ImportBvec	OptimizeDixonEcho	RotationMatrixToQuaternion	TriExponential
CoilWeightedReconCSI	FConvert	ImportDTI	OrderKspace	RotationMatrixToQuaternionVector	UniqueBvalPosi
·	==	_ ' <b>.</b>	'	- ·· ·-	

8 | All-Functions.nb
Options

AcquisitionMethod DixonFrequencies MaskSmoothing ReconFilter ActivationBackground **DixonIterations** MaskWallMap ReferenceB1 ActivationIterations DixonMaskThreshhold MaxSeedPoints RegistrationTarget ActivationOutput DixonNucleus MaxTracts Reject ActivationSize DixonPrecessions MeanMethod RejectMap ActivationThreshold DixonTollerance MeanRes ReportFits AffineDirections DropSamples Method RescaleRecon DropSlices Resolutions AnisoFilterSteps MethodReg EchoShiftData AnisoItterations MethodRegA ResolutionsA AnisoKappa **EPGCalibrate** MonitorCalc ReverseData AnisoKernel EPGFatShift MonitorIVIMCalc ReverseDirection ReversePoints AnisoStepTime **EPGFitFat** MonitorTagging AnisoWeightType **EPGFitPoints** ReverseSets MonitorUnwrap ApodizationFunction **EPGMethod** RobustFit MotionCorrectSets AspectRatio EPGMethodCal NiiDataType RobustFitParameters AxesLabel **EPGRelaxPars** NiiLegacy RotateGradient AxesMethod EPGSmoothB1 NiiMethod RotateGradients NiiOffset RotationCorrect B1EqualPower FatFieldStrength B1FilterData FiberAngle NiiScaling RowSize B1Masking FiberLengthRange NiiSliceCode Runs B1MaxPower FieldStrength NNThreshhold SampleStep B10utput FileType NoiseSize ScaleCorrect B1Scaling FilterMaps NoiseType Scaling B1ShimMethod FilterShape NormalizeDensity SeedDensity BackgroundValue FilterSize NormalizeIVIM SegmentationMethod SelectSubjects BasisSequence FilterType NormalizeMethod BidsIncludeSession FindTransform NormalizeOutputSpectra SenseRescale BinaryType FineTuneFit NormalizeOverlap ShowMetric BloodMaskRange FitConstrains NormalizeSets ShowOutliers BmatrixOut ShowPlot FitFunction NormalizeSignal Boxed SimNucleus FitLineShape NumberSamples BsplineDirections FitOutput NumberSamplesA SliceRange FittingOrder OrderSpan SliceRangeSamples BsplineSpacing BullPlotMethod FixPseudoDiff OutlierIncludeZero SmartMaskOutput FixPseudoDiffSD OutlierIterations SmartMethod CenterFrequency CenterVoxel FlipAxes OutlierMethod SmoothHelix ChainSteps FlipBvec OutlierOutput SmoothItterations CoilArrayPlot FlipGrad OutlierRange SmoothSNR CoilSamples FullOutput OutputCalibration SortVecs CoilSurfaceVoxelSize **FullSphere** OutputCheckImage SparID ColorFunction GetMaskOnly OutputCoilSurface SparName ColorValue GetMaskOutput OutputImage SparOrientation CompressNii GOutput OutputMethod SpectraBandwith ConditionCalc GradType OutputPlot SpectraFieldStrength ContourColor GRegularization OutputSamples SpectraNucleus ContourColorRange GridLines SpectraOutputPlots OutputSense ContourOpacity GridLineSpacing SpectraPpmShift OutputSNR ContourQuality GroupPerSegment OutputTransformation SpectraSamples ContourSize HammingFilter OutputType SpectraSpacing HistogramBins OutputWeights SphereColor ConvertDcm HistogramBinsA CorrectPar Padding SphereSize

UnwrapDimension
UnwrapThresh
UpdateStep
UseGPU
UseGrad
UseMask
UseSubfolders
UseVersion
VersionCheck
VisualOpt
WaterFatShift
WaterFatShiftDirection
WienerRegularization
WindowTitle

UnitMulti

10 | All-Functions.nb In[28]:= QMRIToolsFunctions["All", 10]

# CardiacTools

# Functions BullseyePlot

CalculateWallMap GetSegmentSlices CardiacCoordinateSystem HelixAngleCalc CardiacSegment LinesToSegmentIndex CardiacSegmentGUI MakeECVBloodMask CentralAxes MakeLineImage CreateHeart MakeMaskImage MaskHelix ECVCalc ExcludeSlices MaskToLines GetMaskSegmentPoints PlotSegmentMask

PlotSegments RadialSample SegmentLinesToMask SegmentsPerSlice TransmuralPlot

## **Options**

AxesMethod BackgroundValue BloodMaskRange BullPlotMethod ColorFunction CutOffMethod DistanceMeasure DropSamples	ImageSize LCMMethod MakeSegmentPlots MaskWallMap Method OutputCheckImage PlotLabel PlotRange	ReverseDirection ReversePoints RowSize SegmentationMethod ShowOutliers ShowPlot SmoothHelix StartPoints	TextOffset TextSize
DropSamples	PlotRange	StartPoints	
GridLineSpacing	PlotStyle	StartSlices	
GroupPerSegment	RadialSamples	TextNumberForm	

GetSegmentLines

# CoilTools

#### **Functions**

CoilSNRCalc FindCoilPosition LoadCoilSetup LoadCoilTarget MakeCoilLayout MakeNoisePlots MakeWeightMask

#### **Options**

CoilArrayPlot
CoilSurfaceVoxelSize
ColorFunction
ImageSize
OutputCoilSurface
PlotRange

# DenoiseTools

#### **Functions**

AnisoFilterData
AnisoFilterTensor
DeNoise
DenoiseCSIdata
DenoiseDynamicSpectraData
NNDeNoise
PCADeNoise
PCADeNoiseFit
WeightMapCalc

### **Options**

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

# **DixonTools**

## **Functions**

DixonPhase UnwrapSplit
DixonReconstruct Wrap

DixonToPercent FindInPhaseEchos FixDixonFlips OptimizeDixonEcho SimulateDixonSignal

Unwrap UnwrapDCT UnwrapList

# **Options**DixonAmplitudes

DixonBipolar DixonIterations
DixonClipFraction DixonMaskThreshhold
DixonComplexOutput DixonNucleus
DixonCorrectT1 DixonPrecessions
DixonFieldStrength DixonTollerance
DixonFilterInput MonitorCalc

DixonFrequencies

DixonFilterOutput MonitorUnwrap
DixonFilterSize UnwrapDimension
DixonFilterType UnwrapThresh

#### **Functions**

ReadTransformParameters RegisterCardiacData

RegisterData

RegisterDataSplit

RegisterDataTransform

RegisterDataTransformSplit

RegisterDiffusionData

RegisterDiffusionDataSplit

RegisterTensorData

TransformData

#### **Options**

AffineDirections BsplineDirections BsplineSpacing DeleteTempDirectory FindTransform HistogramBins

HistogramBinsA InterpolationOrderReg

InterpolationOrderRegA Iterations

IterationsA MethodReg MethodRegA

NumberSamples NumberSamplesA OutputImage

OutputTransformation

**PCAComponents** PrintTempDirectory RegistrationTarget

\$debugElastix

Resolutions ResolutionsA ShowMetric SplitMethod TempDirectory

UseGPU

# **FasciculationTools**

#### **Functions**

AnalyzeActivations EvaluateActivation FindActivations SelectActivations

## **Options**

ActivationBackground ActivationIterations ActivationOutput ActivationSize ActivationThreshold IgnoreSlices MaskDilation ThresholdMethod

# **GeneralTools**

#### **Functions**

ApplyCrop
AutoCropData
BSplineCurveFit
ClearTemporaryVariables
CompilebleFunctions
ConvertExtension
CropData
CutData
DataToVector
DecomposeAffineMatrix

DecomposeScaleMatrix
DevideNoZero
DynamicPartition
EmptyDirectoryQ
ExpNoZero
ExtractDemoData
FileSelect
FindCrop

FindMaxDimensions

GetAssetLocation

GridData
GridData3D
GyromagneticRatio
LapFilter
LLeastSquares
LogNoZero
MADNoZero
MakeCoordinates
MakeIntFunction
MeanNoZero

MedFilter
MedianNoZero
MemoryUsage
MonitorCalc
NiiFileExistQ
NNLeastSquares
PadToDimensions
QMRIToolsFuncPrint
QMRIToolsPackages

QuaternionToRotationMatrix
QuaternionVectorToRotationMatrix
RescaleData
ReverseCrop
ReverseDimensions
RMSNoZero
RotateDimensionsLeft
RotateDimensionsRight
RotationMatrixToQuaternion
RotationMatrixToQuaternionVector

SaveImage
SignNoZero
Squeeze
StdFilter
StichData
StringPadInteger
SumOfSquares
TensMat
TensVec
VectorToData

## **Options**

CenterVoxel Padding
CropAlways PadDirection
CropInit PadValue
CropOutput SplineDegree
CropPadding SplineKnotsNumber
FileType SplineRegularization
ImageResolution WindowTitle

Imagekesolution

ImageSize

InterpolationOrder
OutputWeights

# GradientTools

#### **Functions**

BmatrixCorrectGradientsGradBmatrixBmatrixCalcEnergyCalcGradSeqBmatrixConvFinalGradsImportGradObjBmatrixInvFindOrderOverPlusCalcBmatrixRotFullGradSelectBvalueData

UniqueBvalPosition

BmatrixToggle GenerateGradients
CalculateMoments GenerateGradientsGUI
ConditionNumberCalc GetGradientScanOrder

ConvertGrads GetSliceNormal
CorrectBmatrix GetSliceNormalDir

#### **Options**

ConditionCalc PhaseEncoding

FlipAxes Runs
FlipGrad Steps
FullSphere StepSizeI
GradType SwitchAxes
Method UnitMulti
MethodReg UseGrad
OrderSpan VisualOpt

OutputPlot OutputType

# ImportTools

ShiftPar

# **Functions**

BvalRead

GradRead

ReadBrukerDiff

ReadBvalue

ReadDicom

ReadDicomDiff

ReadDicomDir

ReadDicomDirDiff

ReadGradients

ReadVoxSize

## **Options**

BmatrixOut

ConvertDcm

RotateGradient

ScaleCorrect

# **IVIMTools**

### **Functions**

BayesianIVIMFit2 IVIMResiduals BayesianIVIMFit3 MeanBvalueSignal

CorrectParMap ThetaConv FConvert ThetaConvi

FConverti FracCorrect HistogramPar IVIMCalc

IVIMCorrectData
IVIMFunction

## **Options**

ChainSteps IVIMConstrains
CorrectPar IVIMFixed
FilterMaps IVIMTensFit

FilterSize Method

FilterType MonitorIVIMCalc
FitConstrains OutputSamples
FixPseudoDiff Parallelize
FixPseudoDiffSD UpdateStep

IVIMComponents
IVIMConstrained

# **JcouplingTools**

## **Functions**

GetSpinSystem SimReadout
MakeSpinSystem SimRotate
SequencePulseAcquire SimSignal
SequenceSpaceEcho SimSpoil
SequenceSpinEcho SysTable

SequenceSteam
SequenceTSE
SimAddPhase
SimEvolve
SimHamiltonian

#### **Options**

CenterFrequency
FieldStrength
Linewidth
LinewidthShape
ReadoutBandwith
ReadoutMethod
ReadoutOutput
ReadoutPhase
ReadoutSamples
SimNucleus

# LoggingTools

#### **Functions**

AddToLog
CheckFile
DirectoryTree
ExportDirectoryTree
ExportLog
ImportLog
MakeCheckFile
PrintDirectoryTree
ResetLog
ShowLog

#### **Options**

{}

# MaskingTools

### **Functions**

DilateMask
GetMaskData
RescaleSegmentation

HomoginizeData ROIMask
ImportITKLabels SegmentMask
Mask SmoothMask

MaskData SmoothSegmentation MeanSignal SplitSegmentations

MergeSegmentations NormalizeData NormalizeMeanData

### **Options**

GetMaskOnly
GetMaskOutput
MaskClosing
MaskComponents
MaskDilation
MaskFiltKernel
MaskSmoothing
NormalizeMethod
SmoothItterations
UseMask

# MuscleBidsTools

## **Functions**

AddToJson

BidsDcmToNii CheckDataDiscription

ExtractFromJSON
GenerateBidsFileName

GenerateBidsFolderName GenerateBidsName

GenerateBi GetConfig

GetJSONPosition

ImportJSON

MergeJSON

MuscleBidsConvert MuscleBidsMerge MuscleBidsProcess

PartitionBidsFolderName

PartitionBidsName SelectBidsFolders SelectBidsSessions SelectBidsSubjects

ViewConfig

### **Options**

BidsIncludeSession DeleteAfterConversion SelectSubjects VersionCheck

# **NiftiTools**

#### **Functions**

CompressNiiFiles ImportBval MakeNiiOrentationS

CorrectNiiOrientation ImportBvalvec DcmToNii ImportBvec

ExportBmat ImportExploreDTItens

ExportBval ImportNii
ExportBvec ImportNiiDiff
ExportNii ImportNiiDix
ExtractNiiFiles ImportNiiT1
GetNiiOrientation ImportNiiT2

#### **Options**

CompressNii NiiSliceCode
DeleteOutputFolder PositiveZ
FlipBvec RotateGradients

Method UseSubfolders MonitorCalc UseVersion

MonitorCalc NiiDataType NiiLegacy NiiMethod NiiOffset

NiiScaling

PhysiologyTools

#### **Functions**

AlignRespLog ImportPhyslog ImportRespirect PlotPhyslog PlotRespiract

## **Options**

OutputMethod SampleStep

# ${\bf Plotting Tools}$

### **Functions**

ColorFAPlot PlotDefGrid
GetSliceData PlotDuty
GetSlicePositions PlotIVIM
GradientPlot PlotMoments
ListSpherePlot PlotSequence
MakeSliceImages

PlotContour PlotCorrection PlotData PlotData3D

# **Options**

ColorFunction MakeCheckPlot

ContourColor Method

NormalizeIVIM ContourColorRange ContourOpacity PeakNumber ContourQuality PlotColor ContourSize PlotRange DropSlices PlotSpace ImageLegend PositiveZ ImageOrientation SphereColor ImageSize SphereSize

# ProcessingTools

#### **Functions**

B1MapCalc FindOutliers NumberTableForm
B1Shimming FitData ParameterFit
CombineB1 GetMaskMeans ParameterFit2
CorrectJoinSetMotion Hist RotateData
DataTransformation Hist2 RotateTensor

DatTot InvertDataset SetupDataStructure

DatTotXLS JoinSets SmartMask
ErrorPlot MeanRange SNRCalc
FiberDensityMap MeanStd SNRMapCalc
FiberLengths MedCouple SplitSets

#### **Options**

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

# ReconstructionTools

## **Functions**

CoilCombine
CoilWeightedRecon
CoilWeightedReconCSI
DeconvolveCSIdata
FourierKspace2D

FourierKspace3D FourierKspaceCSI FourierRescaleData

FourierShift FourierShifted HammingFilterCSI
HammingFilterData
InverseFourierShift
InverseFourierShifted
MakeHammingFilter

MakeSense MeanType

NoiseCorrelation NoiseCovariance NormalizeSpectra OrderKspace ReadListData SagitalTranspose ShiftedFourier ShiftedInverseFourier

TotalType

### **Options**

ReconFilter RescaleRecon

AcquisitionMethod CoilSamples DeconvolutionMethod EchoShiftData HammingFilter Method NormalizeOutputSpectra OutputSense SenseRescale

WienerRegularization

# RelaxometryTools

### **Functions**

CalibrateEPGT2Fit

TriExponentialT2Fit

CreateT2Dictionary DictionaryMinSearch

EPGSignal EPGT2Fit

NonLinearEPGFit ShiftPulseProfile

T1Fit T1rhoFit T2Fit

# **Options**

DictB1Range EPGMethodCal DictT2fRange **EPGRelaxPars** DictT2fValue EPGSmoothB1

DictT2IncludeWater Method

DictT2Range MonitorCalc

EPGCalibrate OutputCalibration EPGFatShift WaterFatShift

EPGFitFat

EPGFitPoints EPGMethod

WaterFatShiftDirection

# **SimulationTools**

## **Functions**

AddNoise PlotSimulationAngleHist PlotSimulationHist BlochSeries CalculateGfactor PlotSimulationVec

CreateDiffData Pulses ErnstAngle Signal

GESignal SimAngleParameters GetPulseProfile SimParameters GfactorSimulation SimulateDualTR

PlotSimulation SimulateSliceEPG

PlotSimulationAngle Tensor

## **Options**

FatFieldStrength SliceRangeSamples

GOutput SortVecs GRegularization TensOutput

MagnetizationVector

NoiseSize NoiseType PlotRange Reject ReportFits SliceRange

# SpectroTools

#### **Functions**

ApodizeEcho
ApodizeFid
ApodizePadEcho
ApodizePadFid
ApodizePadSpectra
ApodizeSpectra
ChangeDwellTimeFid

ChangeDwellTimeFid
CompareFidFitPlot
CompareSpectraFitPlot
CorrectTEFid

CorrectTESpec
CSIInterface
ExportSparSdat
FindSpectraPpmShift
FitSpectra
FitSpectraResultTable

GetGyro
GetPpmRange
GetSpectraBasisFunctions
GetTimePpmRange

GetTimeRange
ImportSparSdat
MakeSpectraResultPlot
PadEcho
PadFid
PadSpectra
PhaseCorrectSpectra
PhaseShiftSpectra

PlotCSIData

PlotFid

PlotSpectra
ReadjMRUI
ShiftSpectra
SpectraFitResult
TimeShiftEcho
TimeShiftFid
TimeShiftFidV

### **Options**

ApodizationFunction
AspectRatio
BasisSequence
CenterFrequency
FineTuneFit
FitLineShape
GridLines
GridLineSpacing
ImageSize

PlotColor
PlotLabel
PlotLabels
PlotRange
ReadoutType
SparID
SparName
SparOrientation

PaddingFactor

Method

SpectraBandwith
SpectraFieldStrength
SpectraNucleus
SpectraOutputPlots
SpectraPpmShift
SpectraSamples
SpectraSpacing
SplineSpacingFactor

# TaggingTools

#### **Functions**

InitializeFit

AnnalyzeTagging CalculateDispacementParameters

## **Options**

HistoryWeighting MonitorTagging

# **TensorTools**

# **Functions**

ADCCalc EigenvecCalc
AngleCalc ExpTensor
AngleMap FACalc
ConcatenateDiffusionData FlipGradientOrientation

Correct FlipTensorOrientation

Deriv LogTensor
DriftCorrect ParameterCalc
ECalc RemoveIsoImages
EigensysCalc ResidualCalc
EigenvalCalc SigmaCalc

SortDiffusionData

TensorCalc TensorCorrect TransformTensor WestinMeasures

### **Options**

Distribution RobustFit

FilterShape RobustFitParameters
FullOutput RotationCorrect

MeanRes UseMask

Method MonitorCalc NormalizeSignal Parallelize Reject RejectMap

# TractographyTools

#### **Functions**

CombineROIs SelectTractInVol
FiberLength SelectTractPartInVol

FiberTractography SelectTracts

FilterTracts SelectTractTroughPlane FindTensorPermutation SelectTractTroughVol

FitTract TractAngleMap
GetTractValues TractDensityMap
PartTracts TractLengthMap

PlotTracts SeedDensityMap

### **Options**

Boxed StepSize
FiberAngle StopThreshhold
FiberLengthRange TensorFilps

FittingOrder TensorPermutations

ImageSize TracMonitor
InterpolationOrder TractColoring

MaxSeedPoints
MaxTracts
Method

NormalizeDensity

# **VisteTools**

#### **Functions**

DatRead DatWrite DTItoolExp

DTItoolExpFile

DTItoolExpInd

 ${\tt DTItoolExpTens}$ 

ExportVol

ImportDTI
ImportVol

LoadFiberTracts

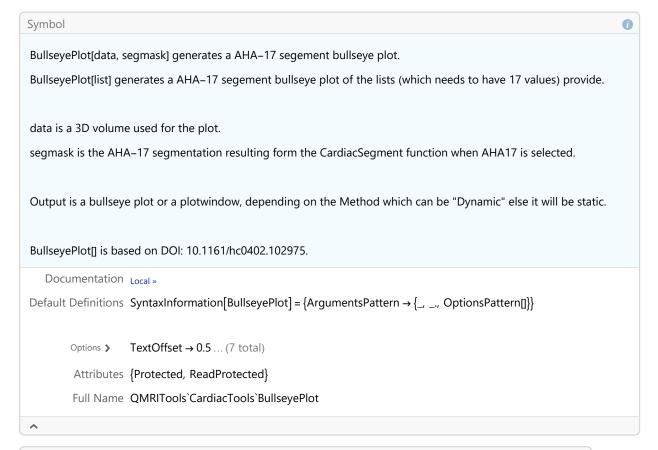
### **Options**

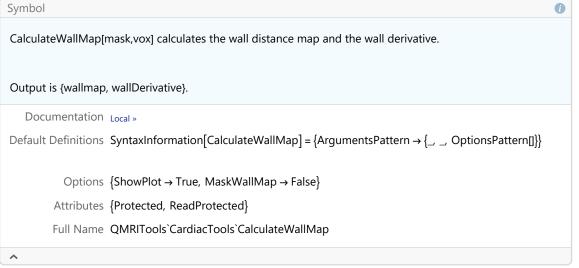
BinaryType

30 | All-Functions.nb

# CardiacTools

#### **Functions**





```
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}.

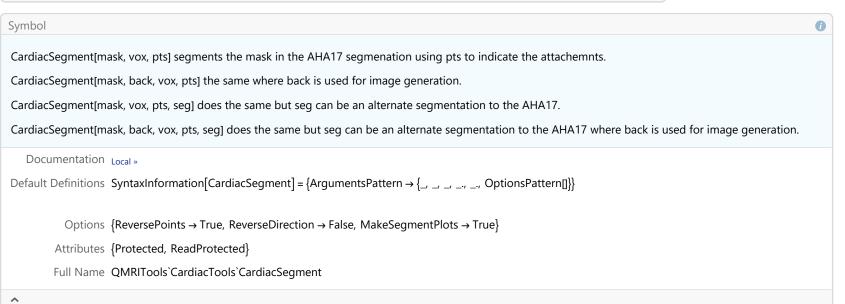
Documentation Local >>

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

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

Attributes {Protected, ReadProtected}

Full Name QMRITools`CardiacTools`CardiacCoordinateSystem
```



CentralAxes[mask, vox] calculates the center of the lumen from a mask of the left ventricle. vox is the voxels size, {slice, x, y}.

CentralAxes[mask, maskp, vox] allows for fancy visualization of the other structures using maskp.

Output is {centerpoints, normalvecs, inout} or {centerpoints, normalvecs, inout, fit}.

Documentation Local >>

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

Options {ShowPlot → False, RowSize → Automatic, AxesMethod → Cubic}

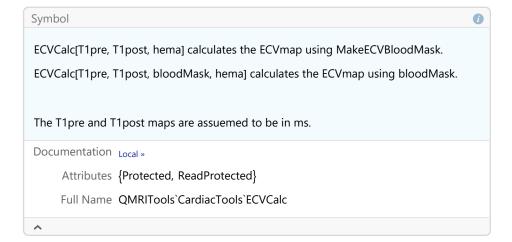
Attributes {Protected, ReadProtected}

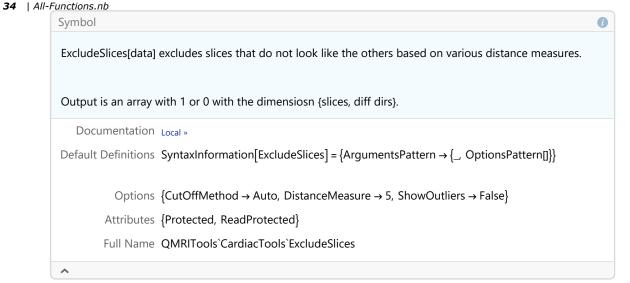
Full Name QMRITools CardiacTools CentralAxes

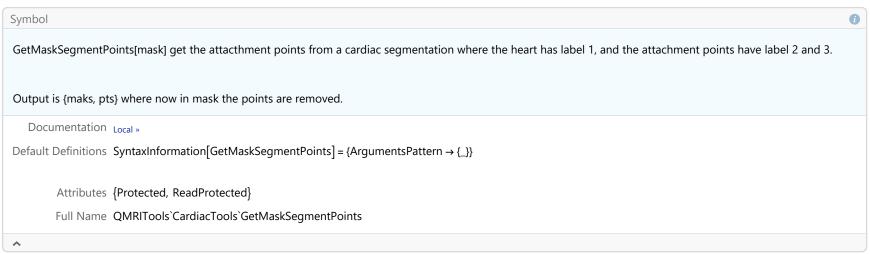
Full Name QMRITools`CardiacTools`CardiacSegmentGUI

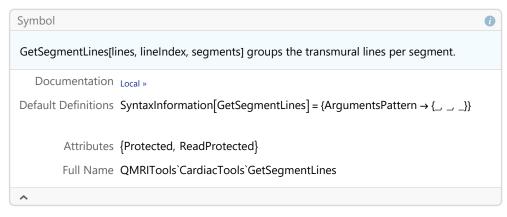
 $\wedge$ 

All-Functions.nb | 33









^

All-Functions.nb | 35

#### Symbol

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

Symbol

LinesToSegmentIndex[lines, points, segments] finds the lines indeces correspoinding to the points and the segments borders. Additionally it finds all the lines indeces for all lines within each segment.

The lines are comupted by MaskToLines, the points are cumputed by GetMaskSegmentPoints, and the segments is the output of SegmentsPerSlices.

Output {pointIndex, segmentIndex}.

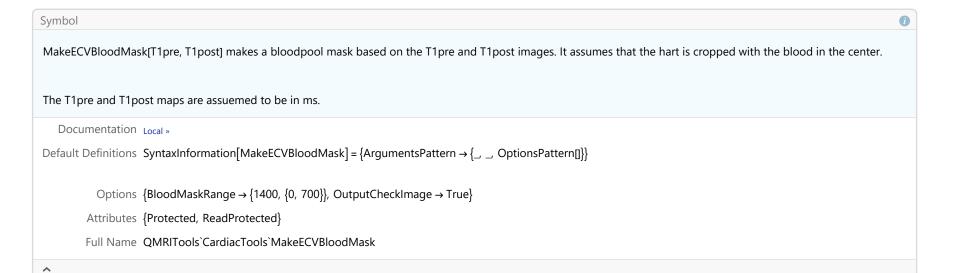
Documentation Local >

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

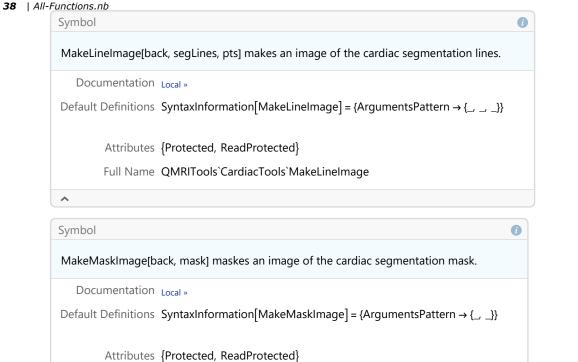
Options {ReversePoints → True, ReverseDirection → False}

Attributes {Protected, ReadProtected}

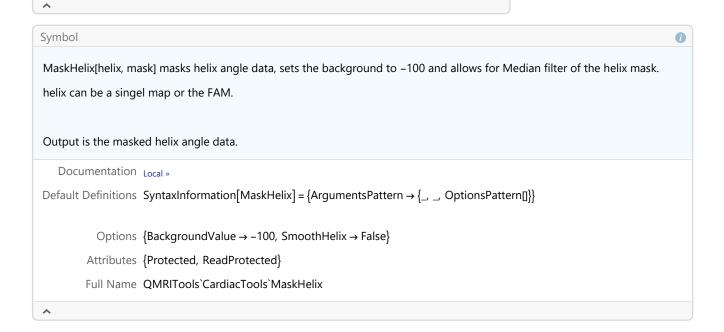
Full Name QMRITools'CardiacTools'LinesToSegmentIndex



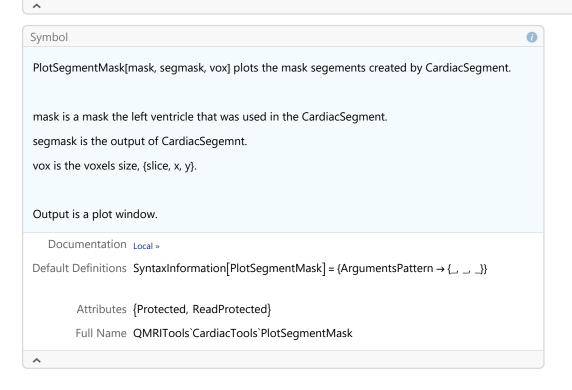
^

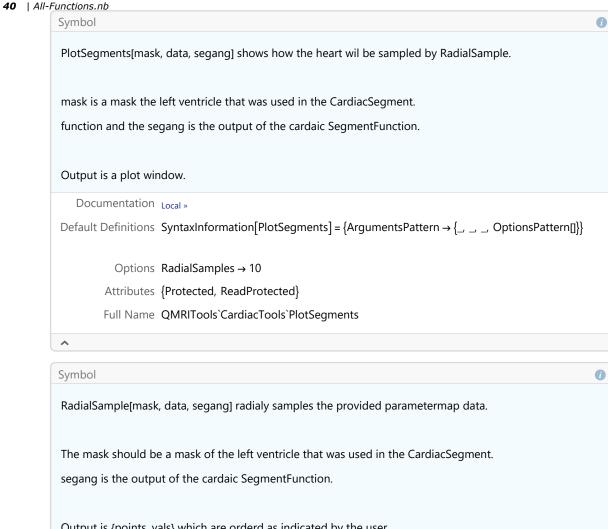


Full Name QMRITools`CardiacTools`MakeMaskImage



Symbol 0 MaskToLines[mask, vox] calculates lines perpendicular to the heart wall per slice within the mask. Internally it uses CalculateWallMap and CentralAxes to obtain the cardiac geometry from mask. MaskToLines[mask, wall, cent] where mask is the first output of CalculateWallMap and cent is the first output of CentralAxes. Documentation Local » Default Definitions SyntaxInformation[MaskToLines] =  $\{ArgumentsPattern \rightarrow \{\_, \_, \_\}\}$ Attributes {Protected, ReadProtected} Full Name QMRITools`CardiacTools`MaskToLines





```
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
```

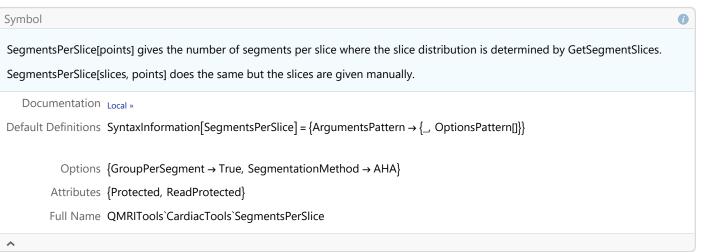
```
SegmentLinesToMask[mask, segLines] cuts the mask based one the tranmural lines per segments which can be obtained by GetGesmentLines.

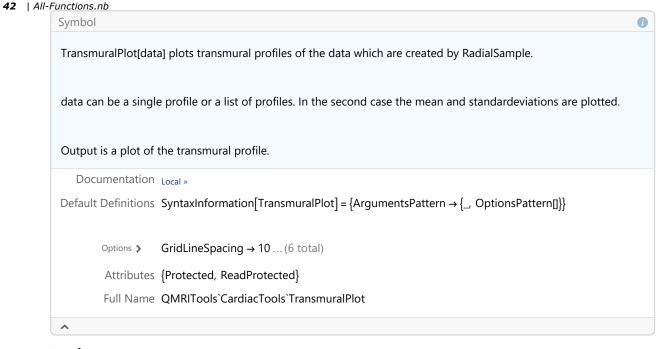
Documentation Local »

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

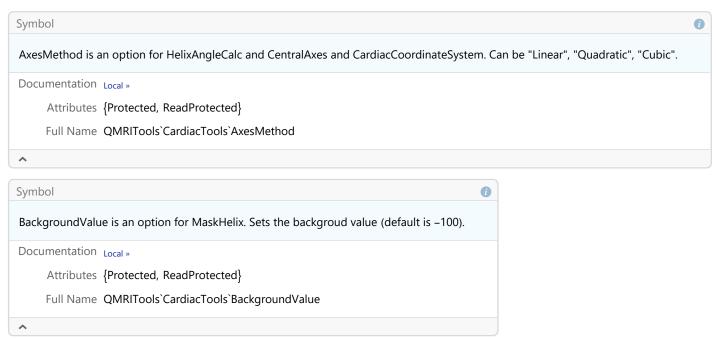
Attributes {Protected, ReadProtected}

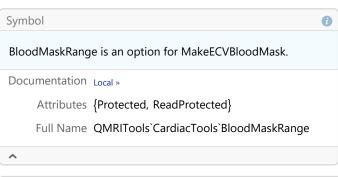
Full Name QMRITools'CardiacTools'SegmentLinesToMask
```

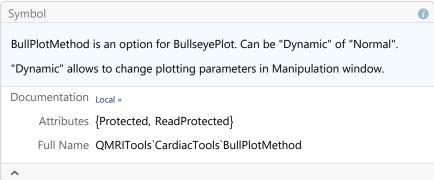


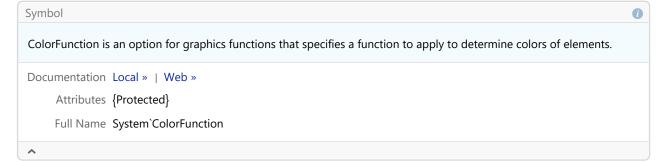


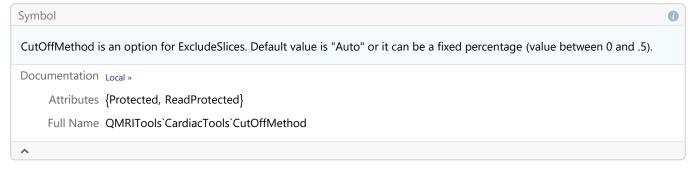
#### **Options**

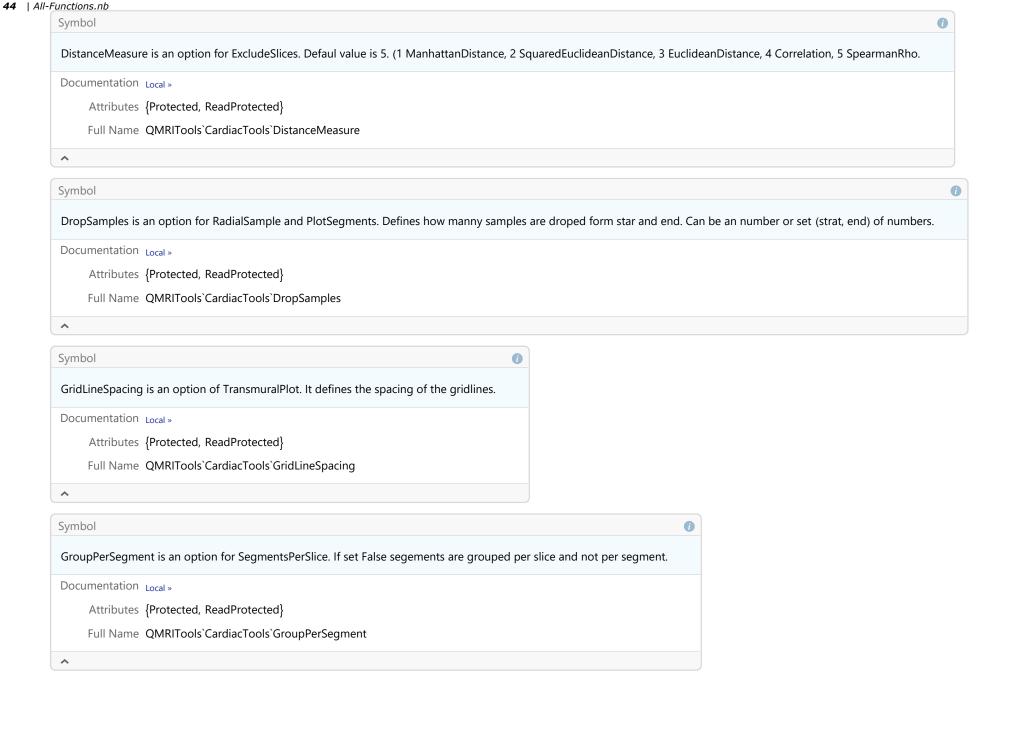




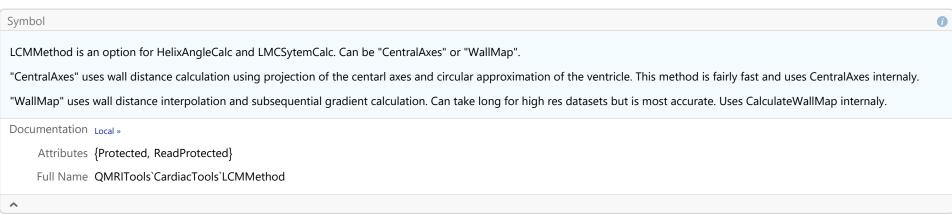


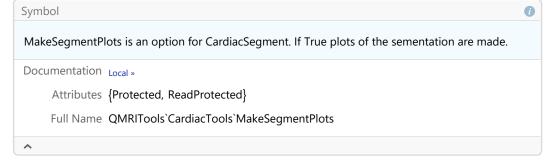


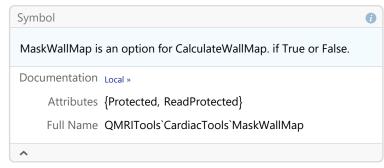


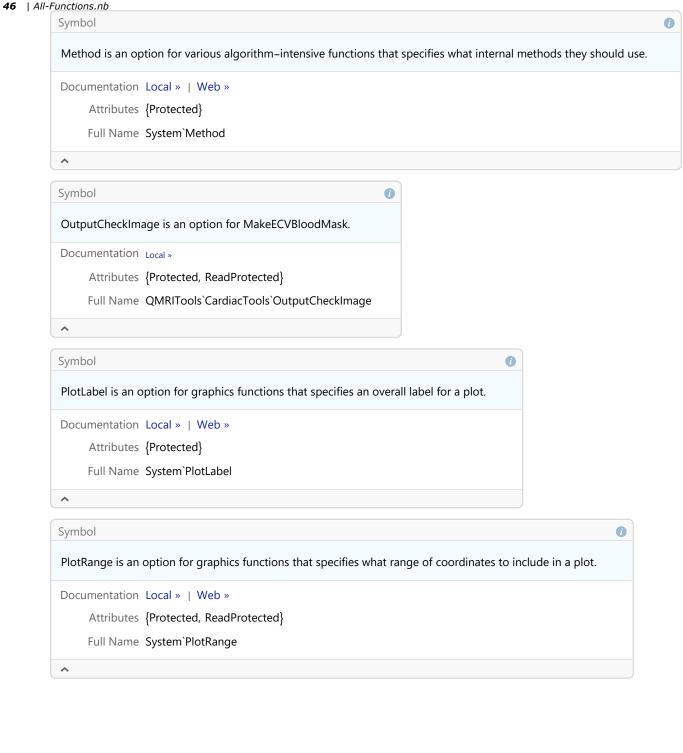


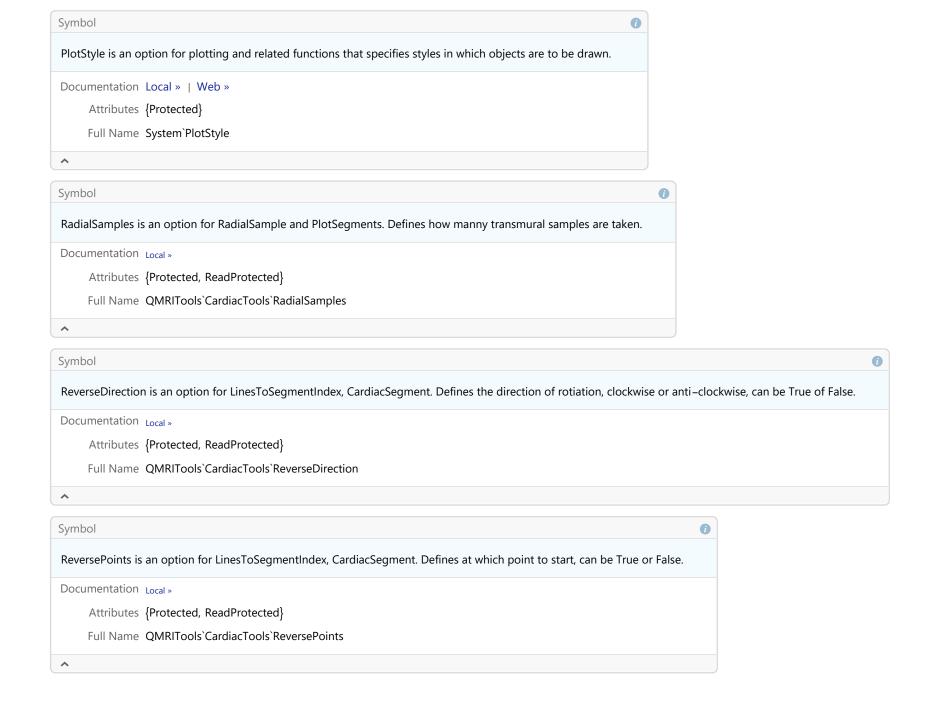


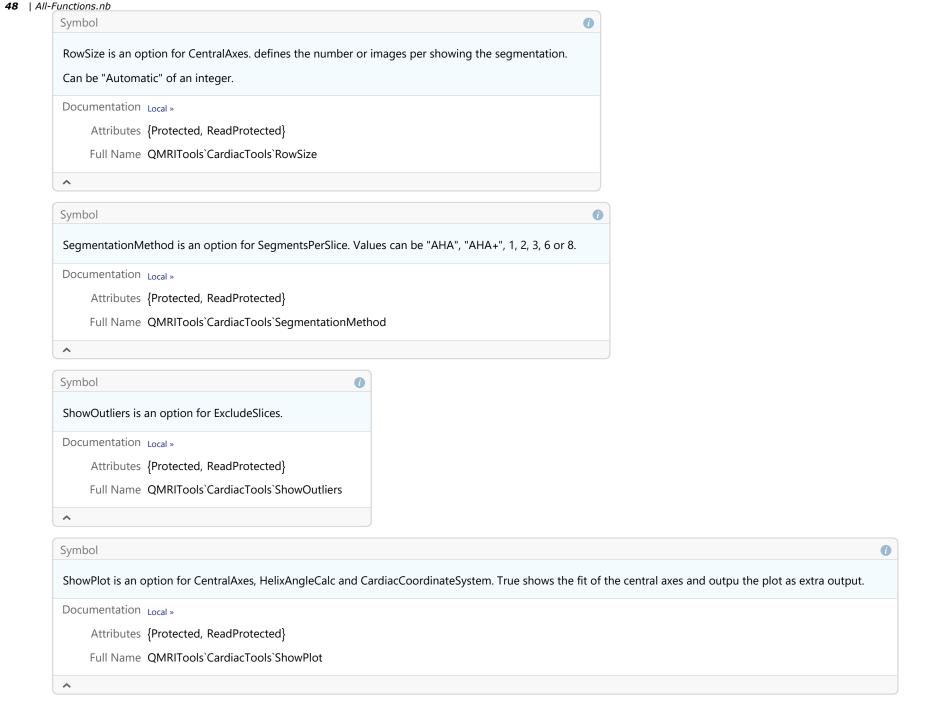


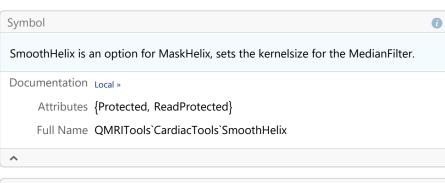


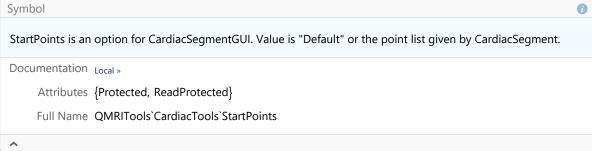




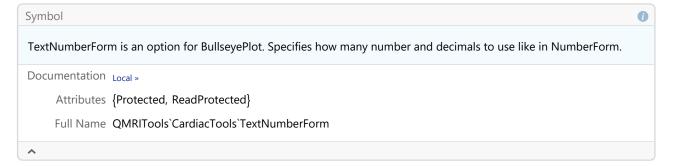


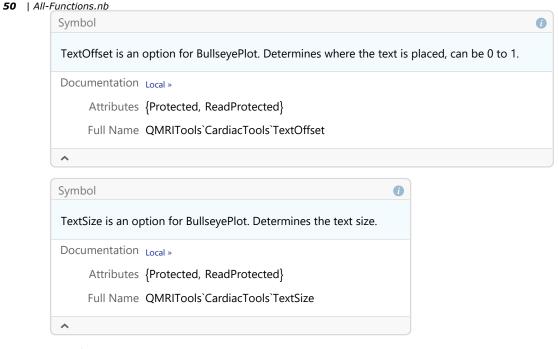






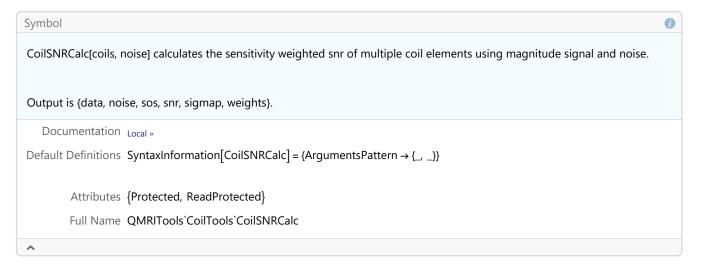






# CoilTools

#### **Functions**



```
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.

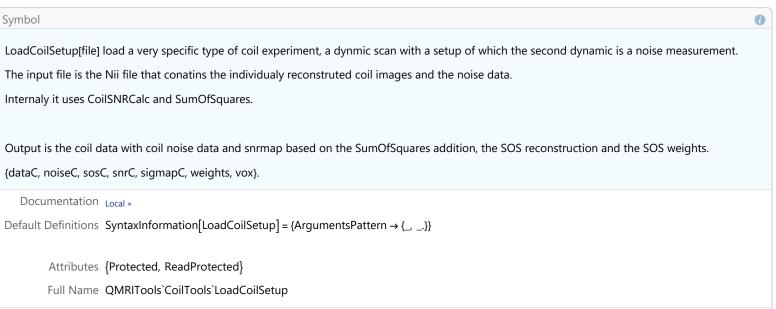
Documentation Local →

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

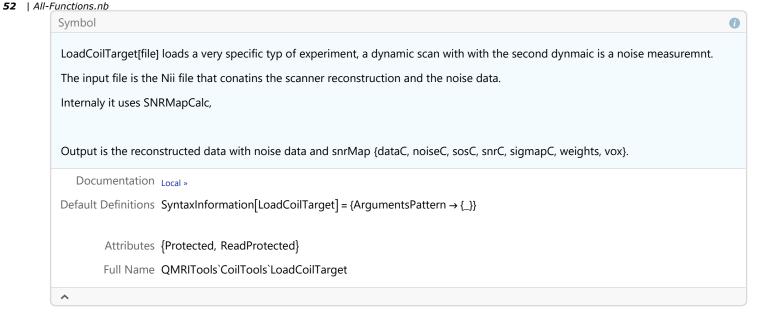
Options {OutputCoilSurface → False, CoilSurfaceVoxelSize → {1, 1, 1}}

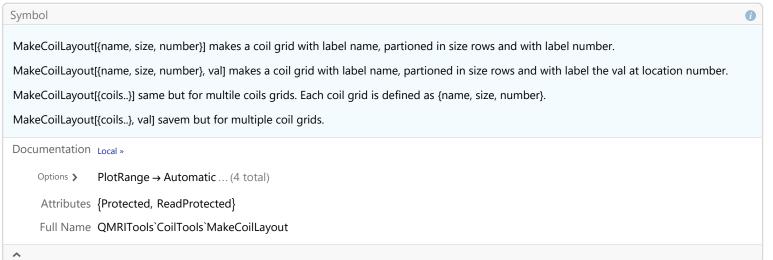
Attributes {Protected, ReadProtected}

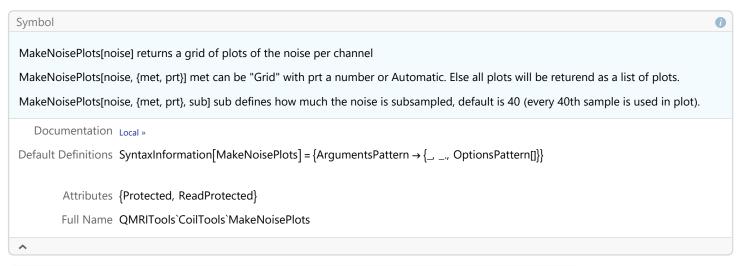
Full Name QMRITools'CoilTools'FindCoilPosition
```

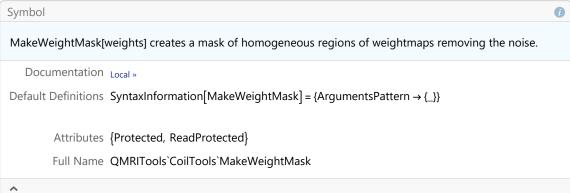


^

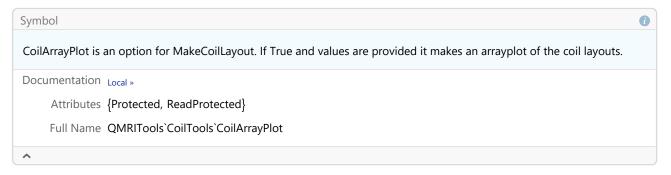


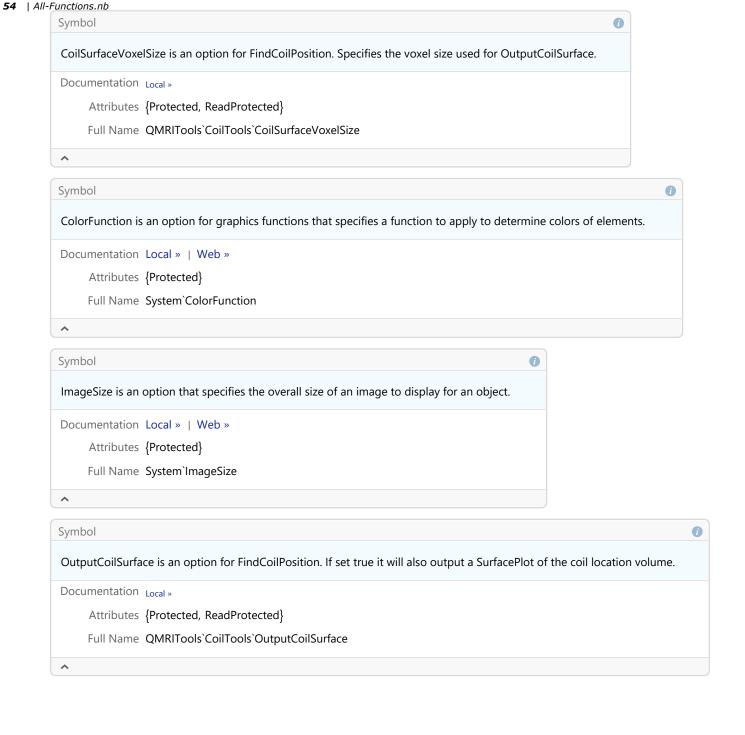






#### **Options**





```
PlotRange is an option for graphics functions that specifies what range of coordinates to include in a plot.

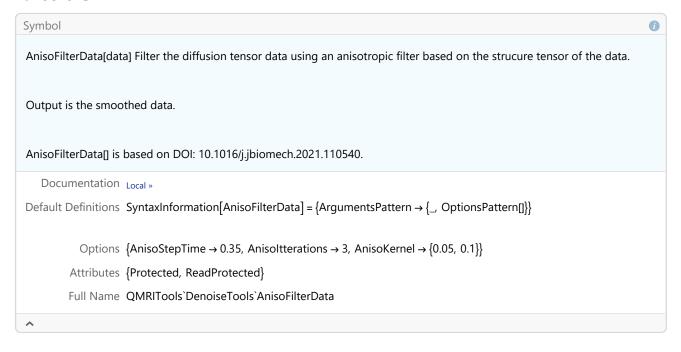
Documentation Local » | Web »

Attributes {Protected, ReadProtected}

Full Name System`PlotRange
```

# DenoiseTools

#### **Functions**



**56** | All-Functions.nb Symbol AnisoFilterTensor[tens, diffdata] Filter the tensor tens using an anisotropic diffusion filter (Pero 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 → {\_\_ \_ \_ OptionsFilterTensor} AnisoWeightType  $\rightarrow 2 \dots (4 \text{ total})$ Options > Attributes {Protected, ReadProtected} Full Name QMRITools`DenoiseTools`AnisoFilterTensor

	(i)
na–Malik).	
_ ,,	
Pattern[]}}	

DenoiseCSIdata[spectra] perfroms PCA denoising of the complex values spectra, data has to be 3D and the spectral dimensions is last, {x,y,z,spectra}.

Documentation Local >

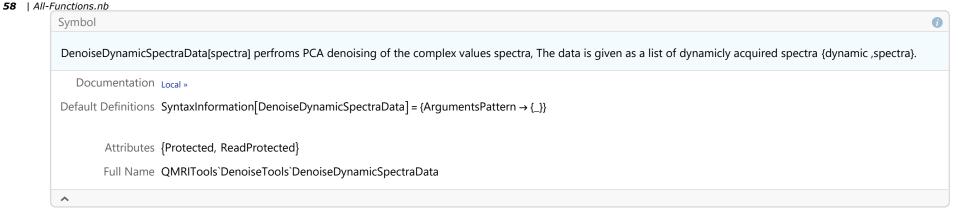
Default Definitions SyntaxInformation[DenoiseCSIdata] = {ArgumentsPattern → {\_, OptionsPattern[]}}

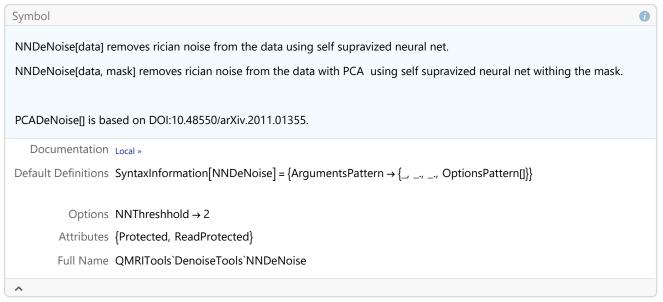
Options {PCAKernel → 5, PCANoiseSigma → Corners}

Attributes {Protected, ReadProtected}

Full Name QMRITools`DenoiseTools`DenoiseCSIdata

 $\wedge$ 





PCADeNoiseFit[data] fits the marchencopasteur distribution to the PCA of the data using grid search.

PCADeNoiseFit[data, sig] fits the marchencopasteur distribution to the PCA of the data using sig as start value or fixed value using grid search.

Output is {simga, number of noise comp, and denoised matrix}.

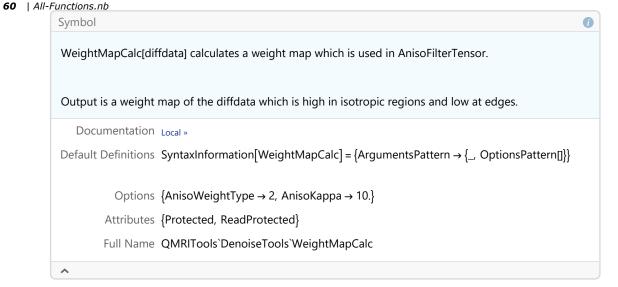
Documentation Local »

Default Definitions SyntaxInformation[PCADeNoiseFit] = {ArgumentsPattern → { \_ \_ \_ \_ \_ \_ \_ \_ \_ }}

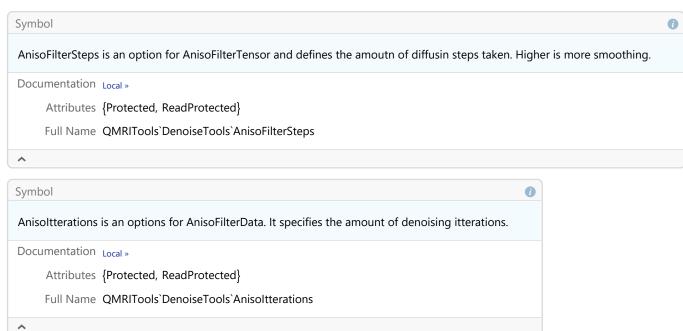
Attributes {Protected, ReadProtected}

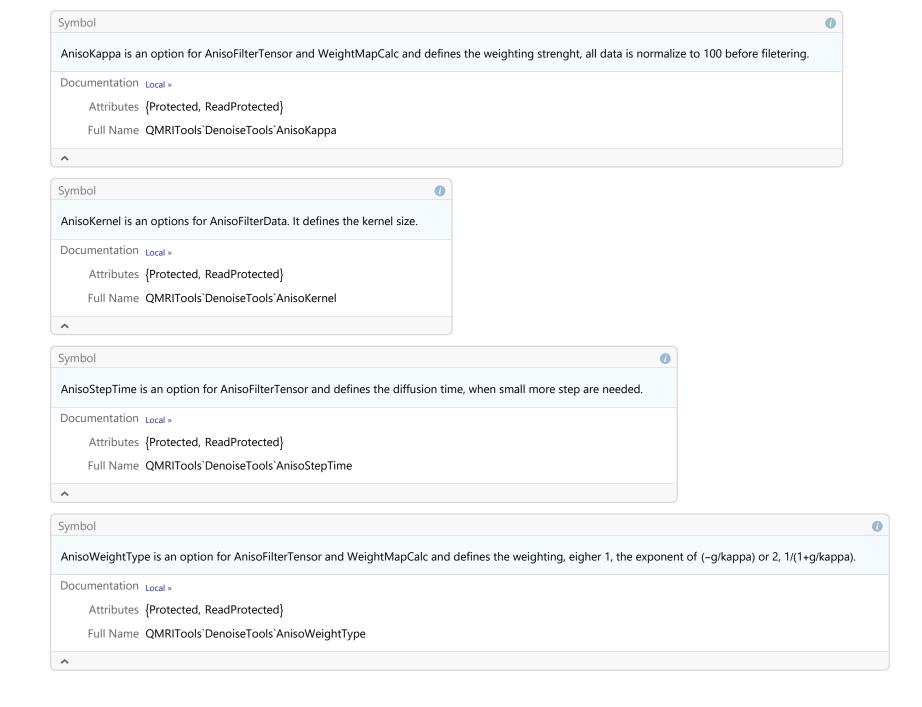
Full Name QMRITools'DenoiseTools'PCADeNoiseFit

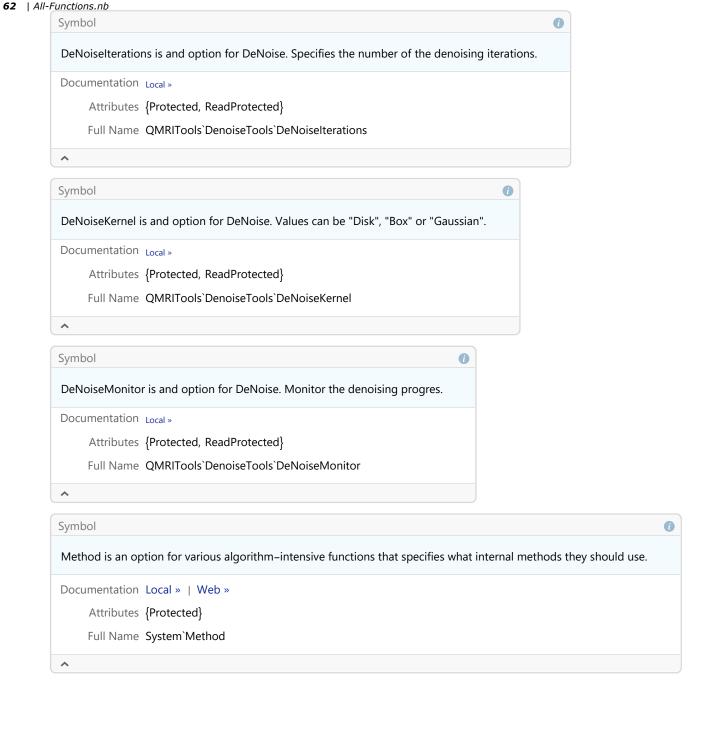
Full Name QMRITools`DenoiseTools`PCADeNoise

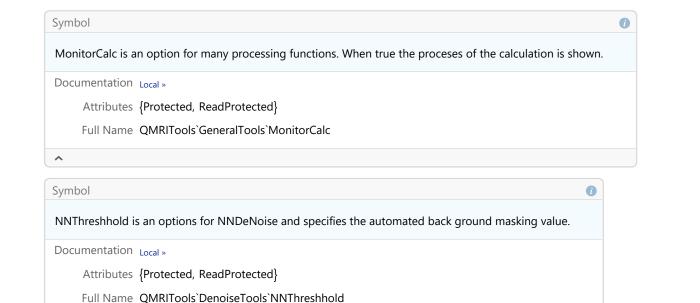


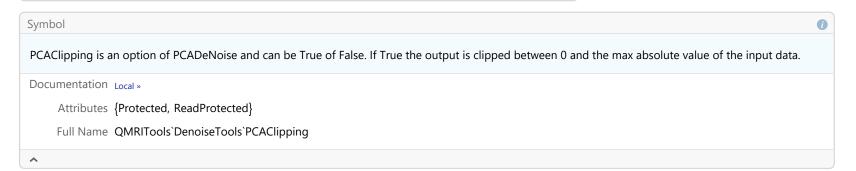
### **Options**

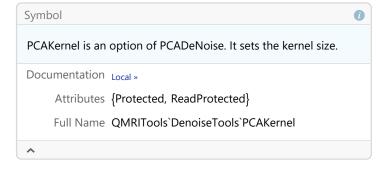




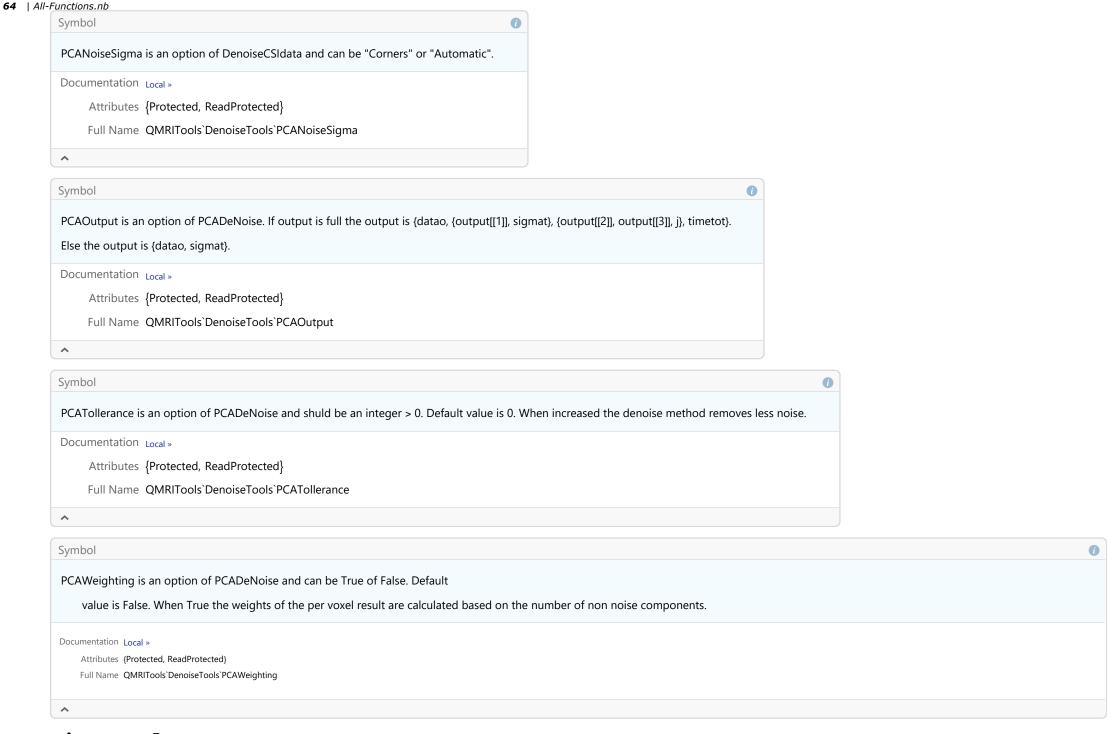








^



## **DixonTools**

#### **Functions**

```
DixonPhase[real, imag, echos] calculates the b0 and ph0 maps.

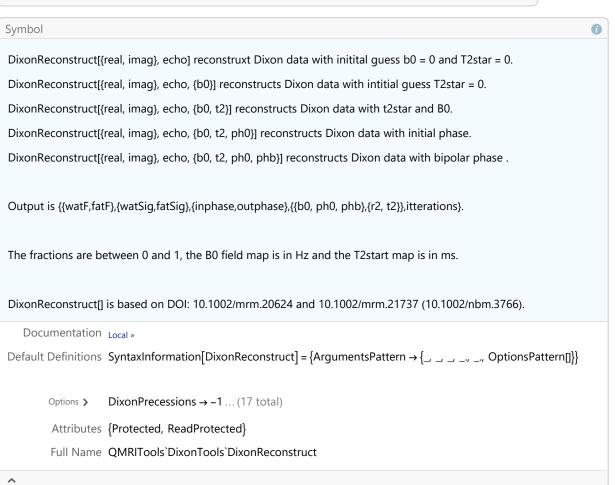
Documentation Local »

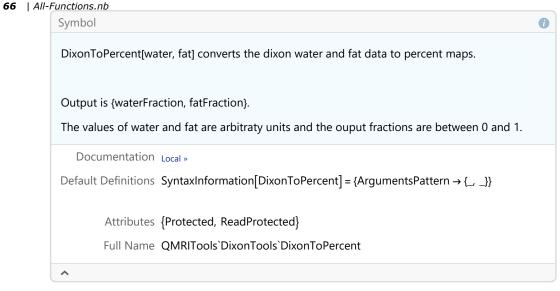
Default Definitions SyntaxInformation[DixonPhase] = {ArgumentsPattern → {_, _, OptionsPattern[]}}

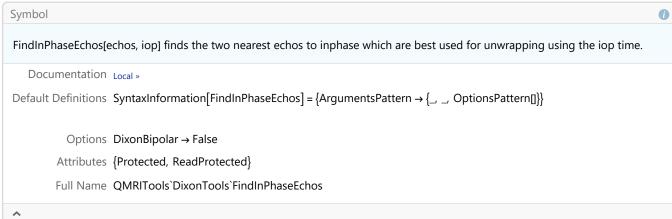
Options ➤ DixonBipolar → True ... (7 total)

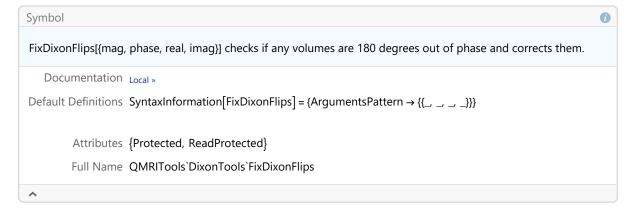
Attributes {Protected, ReadProtected}

Full Name QMRITools`DixonTools`DixonPhase
```









```
OptimizeDixonEcho[] shows a manipulate pannel which allos to optimize the dixon echos.

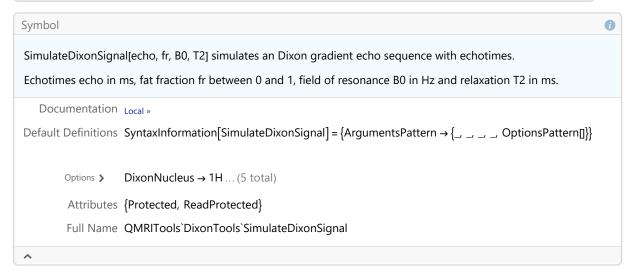
OptimizeDixonEcho[echos] shows a manipulate pannel which allos to optimize the predifined dixon echos.

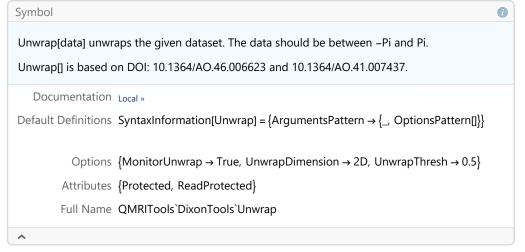
Documentation Local »

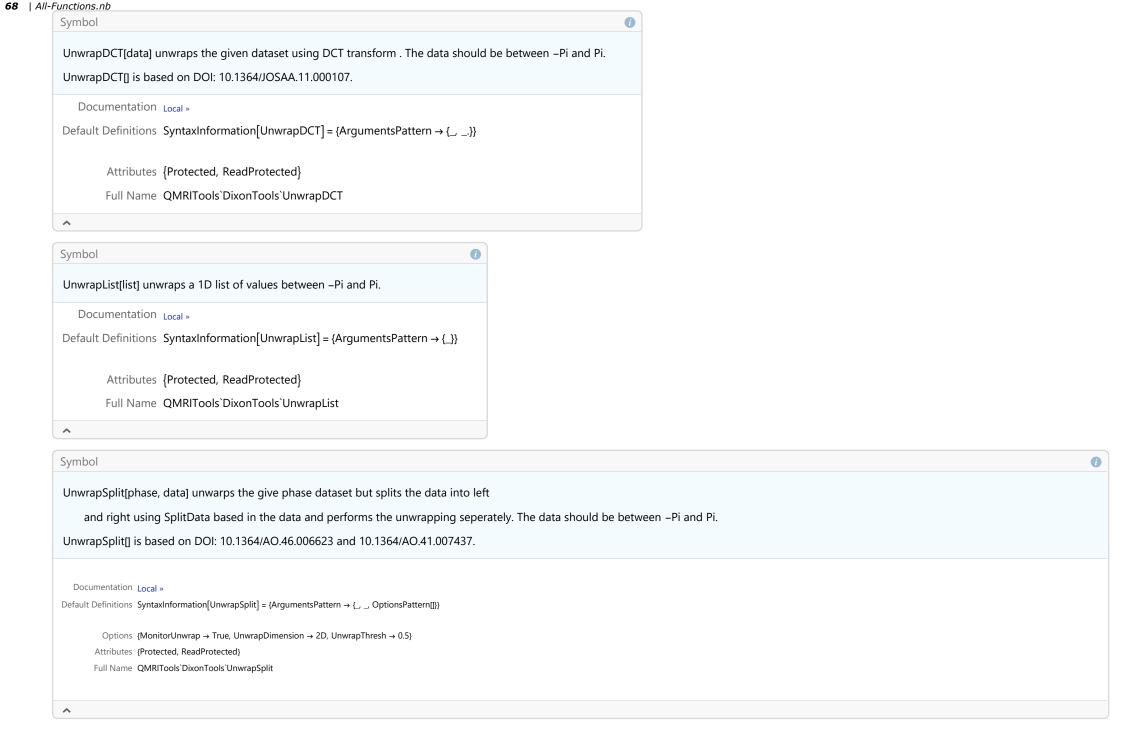
Options ➤ DixonPrecessions → -1 ... (5 total)

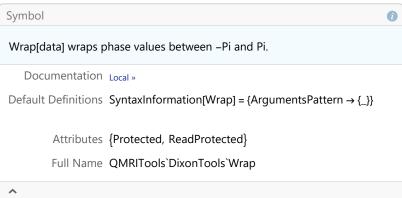
Attributes {Protected, ReadProtected}

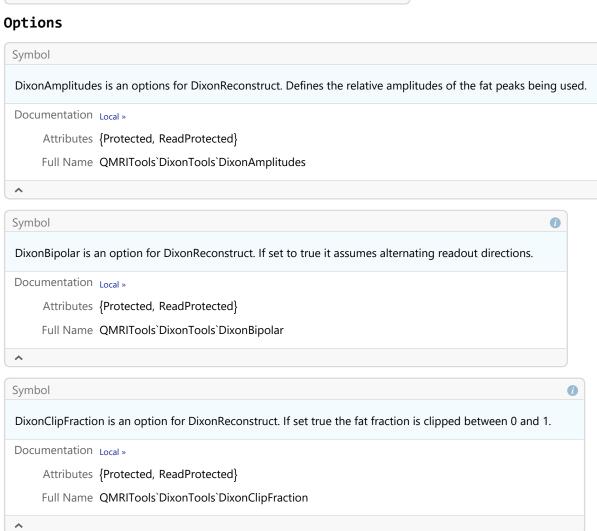
Full Name QMRITools`DixonTools`OptimizeDixonEcho
```

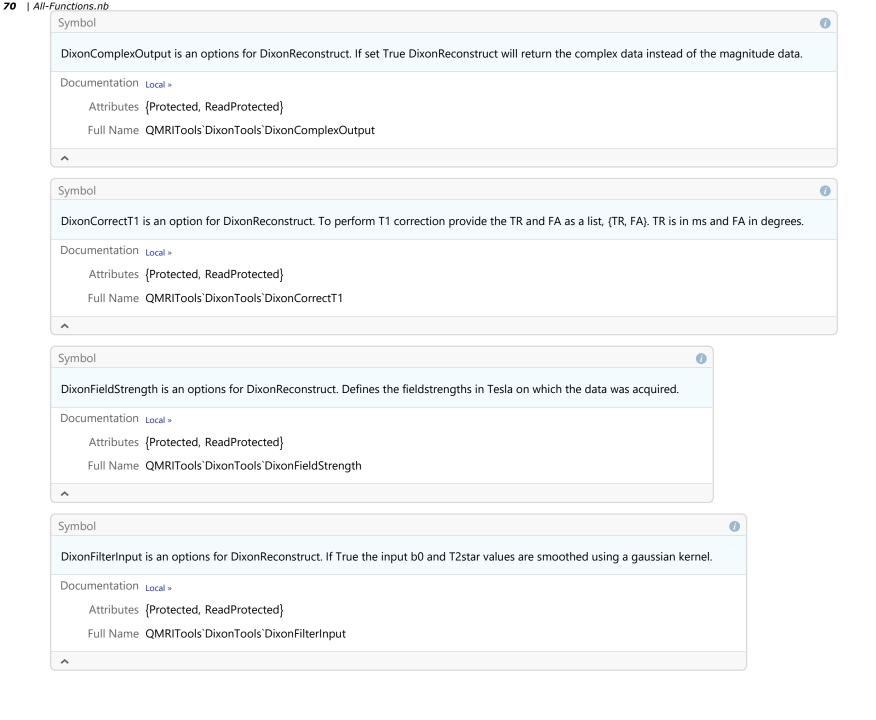


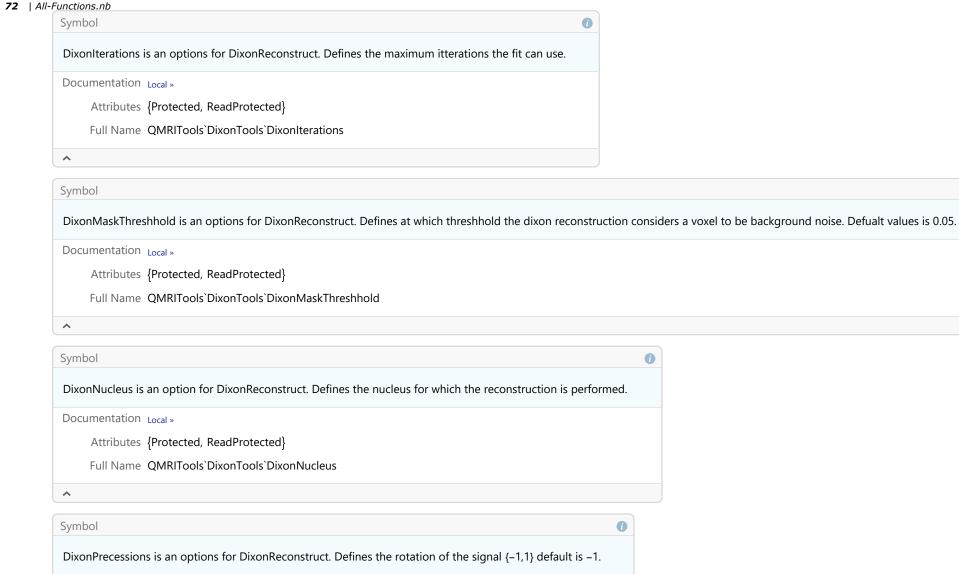










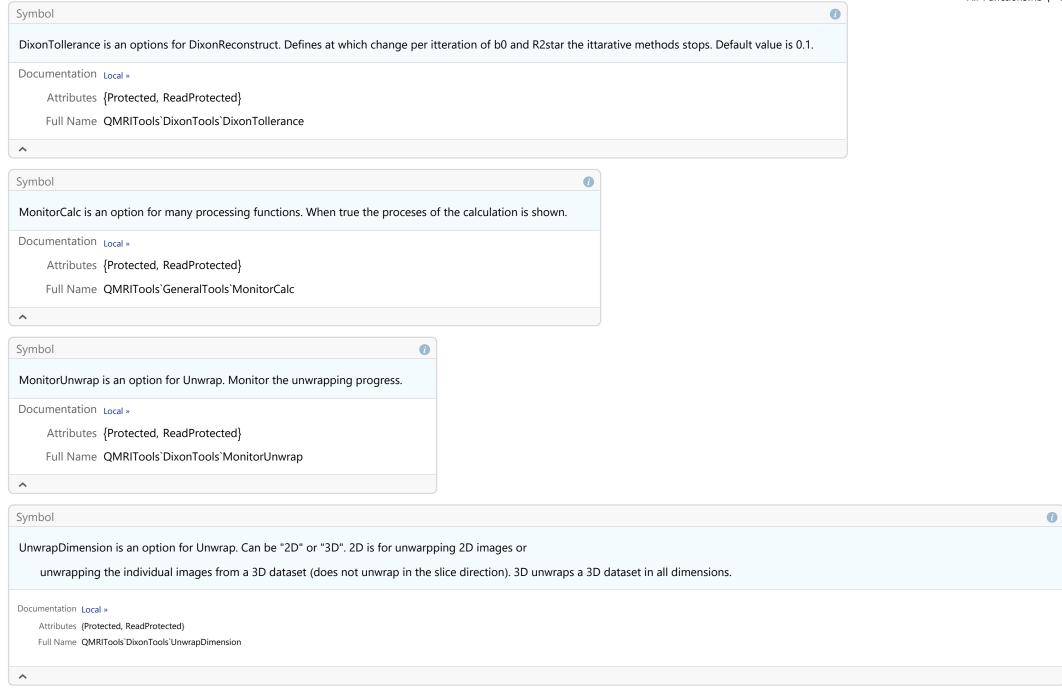


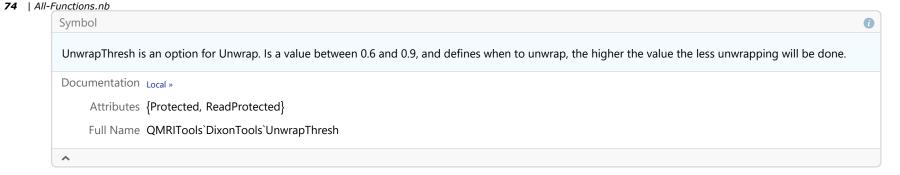
Documentation Local »

 $\wedge$ 

Attributes {Protected, ReadProtected}

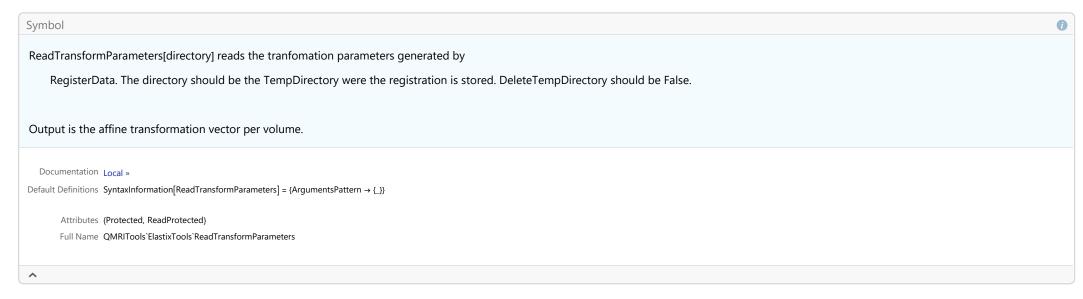
Full Name QMRITools`DixonTools`DixonPrecessions





# **ElastixTools**

#### **Functions**



All-Functions.nb | 75

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  $\rightarrow$  {\_, OptionsPattern[]}}

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

Attributes {Protected, ReadProtected}

Full Name QMRITools`ElastixTools`RegisterCardiacData

 $\wedge$ 

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 »

 $\label{eq:definitions} Default Definitions \ SyntaxInformation[RegisterData] = \{ArgumentsPattern \rightarrow \{\_,\_,OptionsPattern[]\}\}$ 

Options  $\rightarrow$  Iterations  $\rightarrow$  250 ... (17 total)

Attributes {Protected, ReadProtected}

Full Name QMRITools`ElastixTools`RegisterData

^

Symbol

RegisterDataSplit[target, moving] is identical to RegisterData data however left and right side of the data are registered seperately.

Splitting the data is done using the function CutData and merged wit Stich data.

Output is the registered data.

Documentation Local »

 $\label{eq:definitions} Default \ Definitions \ \ SyntaxInformation[RegisterDataSplit] = \{ArgumentsPattern \rightarrow \{\_,\_,\_, \ OptionsPattern[]\}\}$ 

Options  $\Rightarrow$  Iterations  $\Rightarrow$  250 ... (18 total)

Attributes {Protected, ReadProtected}

Full Name QMRITools`ElastixTools`RegisterDataSplit

^

0

Symbol

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 according to the output of the registration 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 »

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

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

Attributes {Protected, ReadProtected}

Full Name QMRITools`ElastixTools`RegisterDataTransform

Symbol

 $\wedge$ 

^

•

RegisterDataTransformSplit[target, moving, {moving2nd, vox}] is idendited 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 »

 $\label{eq:definitions} Default \ Definitions \ \ SyntaxInformation[RegisterDataTransformSplit] = \\ \left\{ ArgumentsPattern \rightarrow \left\{ \_, \_, \_, OptionsPattern[] \right\} \right\}$ 

Options  $\rightarrow$  Iterations  $\rightarrow$  250 ... (18 total)

Attributes {Protected, ReadProtected}

Full Name QMRITools`ElastixTools`RegisterDataTransformSplit

**78** | All-Functions.nb

Symbol

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 »

 $\wedge$ 

 $Default\ Definitions\ \ SyntaxInformation[RegisterDiffusionData] = \{ArgumentsPattern \rightarrow \{\_\_.,\ OptionsPattern[]\}\}$ 

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

Attributes {Protected, ReadProtected}

Full Name QMRITools`ElastixTools`RegisterDiffusionData

RegisterDiffusionDataSplit[dtidata, vox] is identical to Register diffusion data however left and right side of the data are registered seperately.

RegisterDiffusionDataSplit[dtidata, vox), {anatdata, voxa}] is identical to Register diffusion data however left and right side of the data are registered seperately.

RegisterDiffusionDataSplit[dtidata, dtimask, vox), {anatdata, anatmask, voxa}] is identical to Register diffusion data however left and right side of the data are registered seperately.

Splitting the data is done using the function CutData and merged wit Stich data.

Output is the registered data.

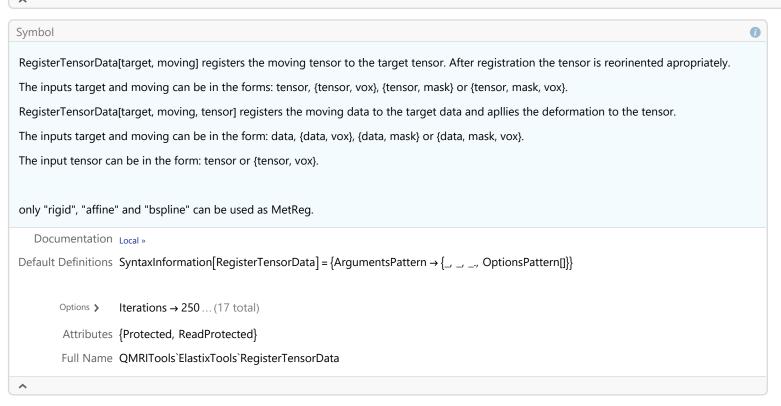
Documentation Local \*\*

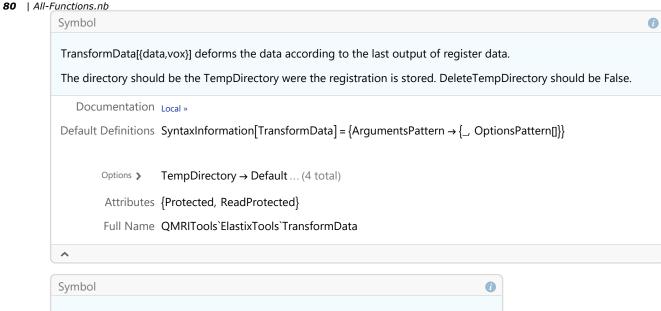
Default Definitions Options[RegisterDiffusionDataSplit] := Options[RegisterDiffusionData]

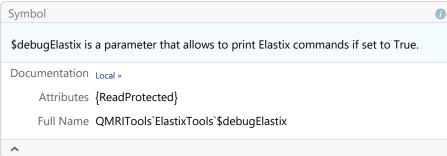
SyntaxInformation[RegisterDiffusionDataSplit] = {ArgumentsPattern → { \_\_ \_\_, OptionsPattern[]}}

Attributes {Protected, ReadProtected}

Full Name QMRITools'ElastixTools'RegisterDiffusionDataSplit







## **Options**

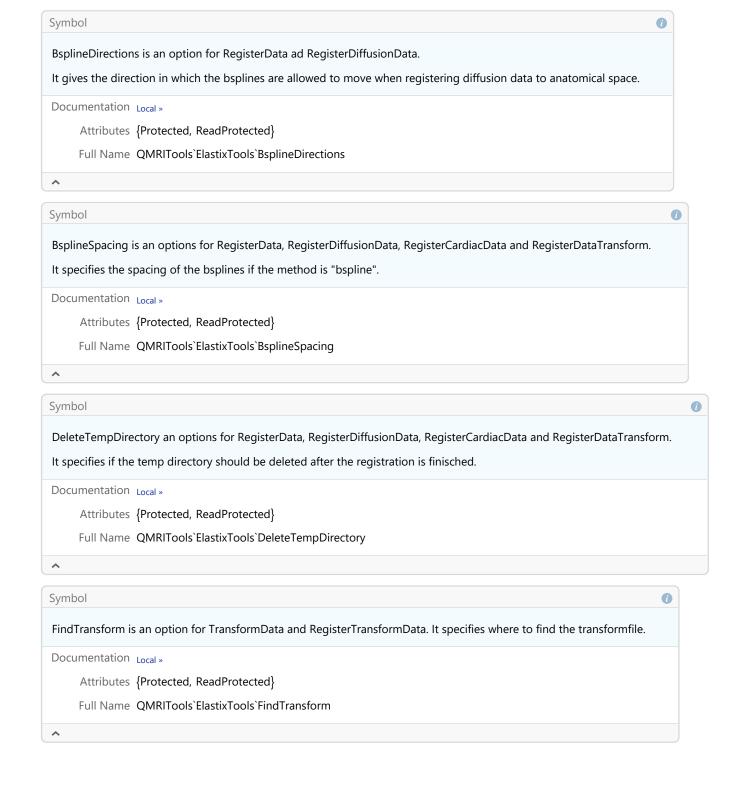
```
AffineDirections is an option for RegisterData ad RegisterDiffusionData.

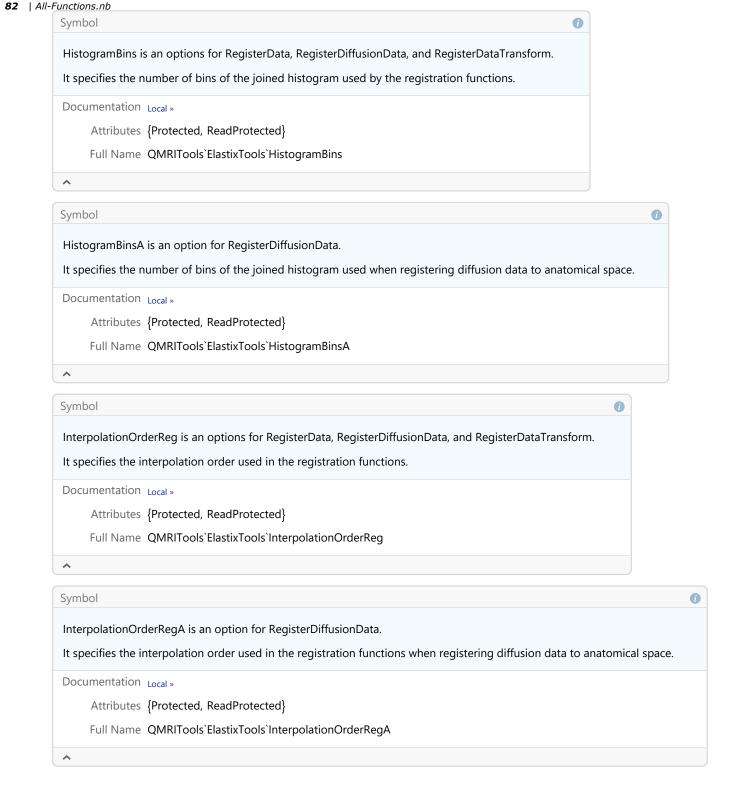
It gives the directions in which data can be moved when registering diffusion data to anatomical space.

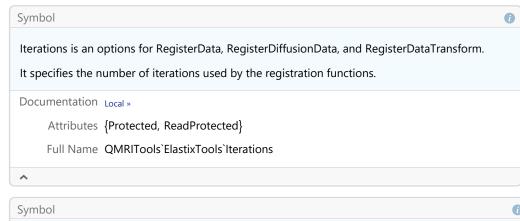
Documentation Local »

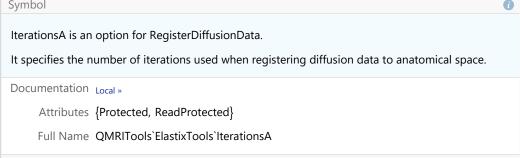
Attributes {Protected, ReadProtected}

Full Name QMRITools`ElastixTools`AffineDirections
```

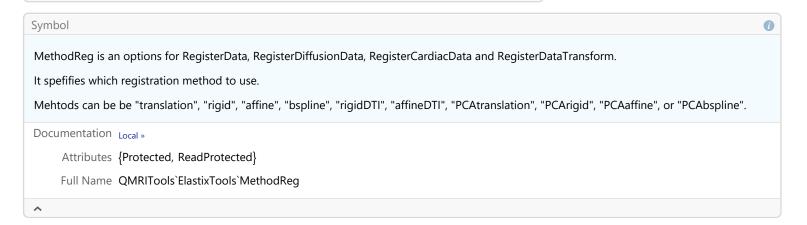


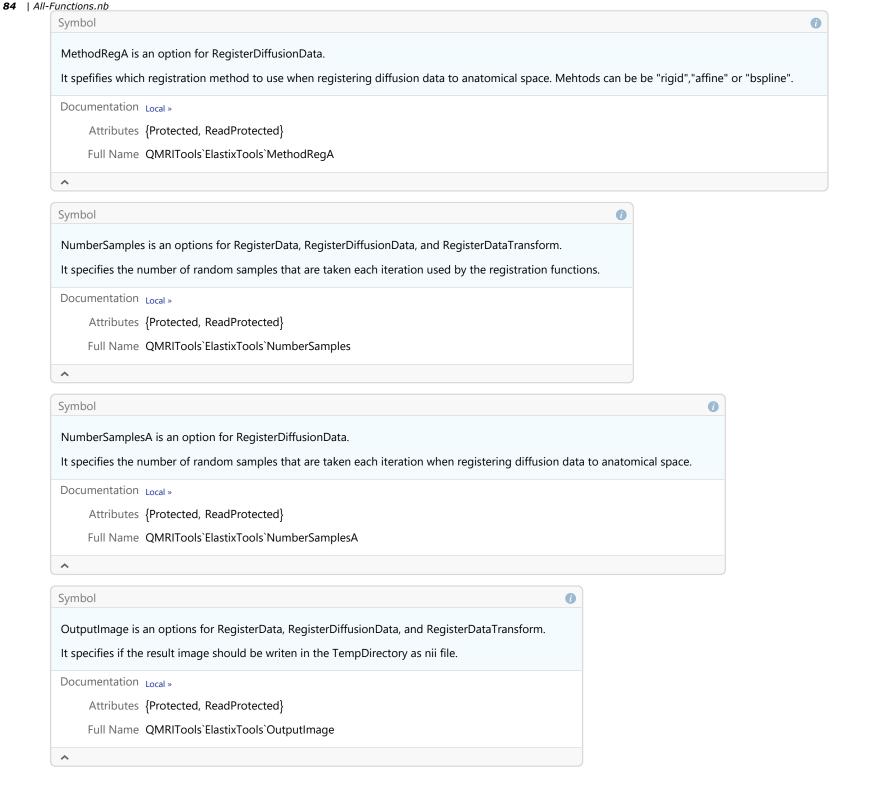


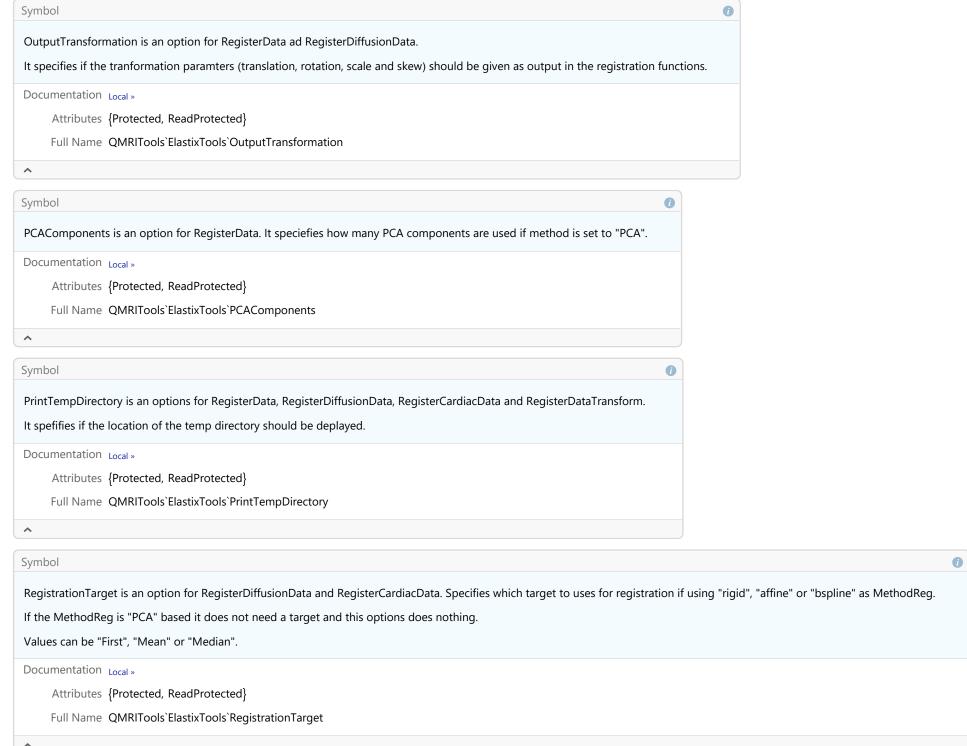


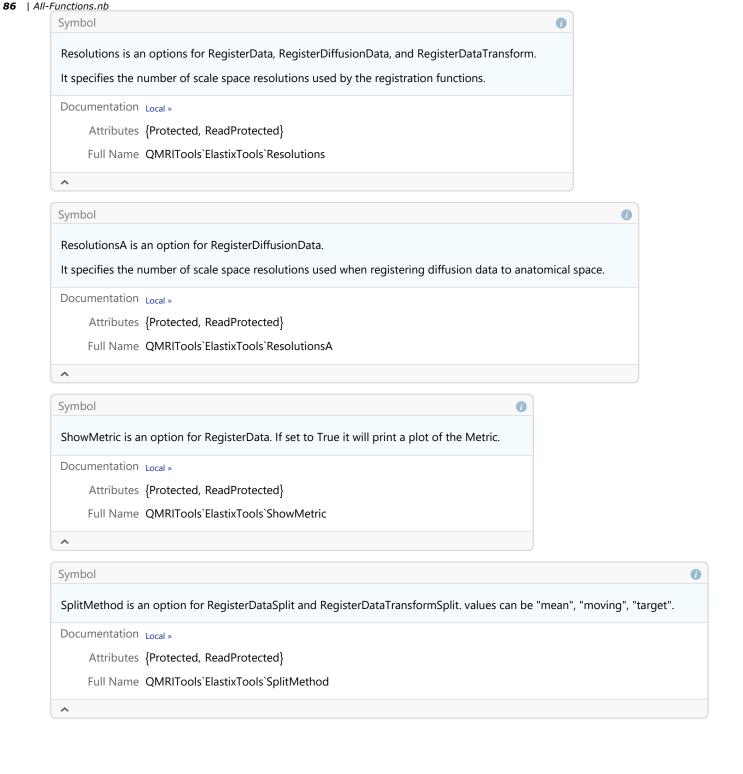


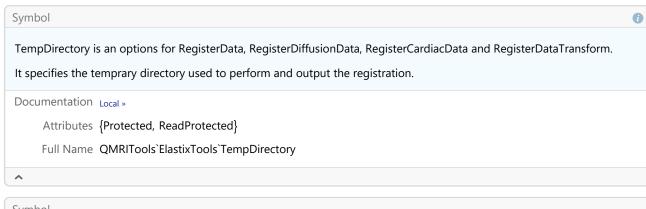
 $\wedge$ 

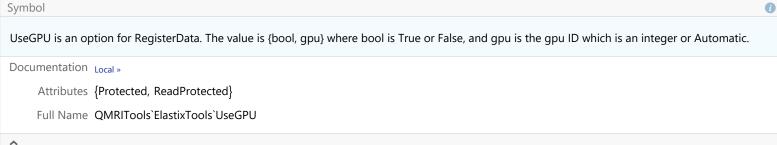






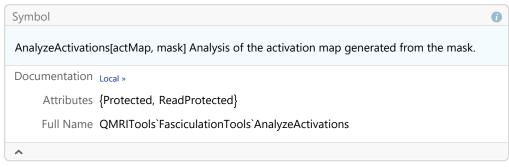


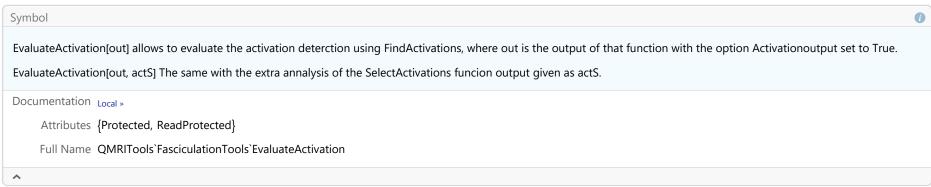


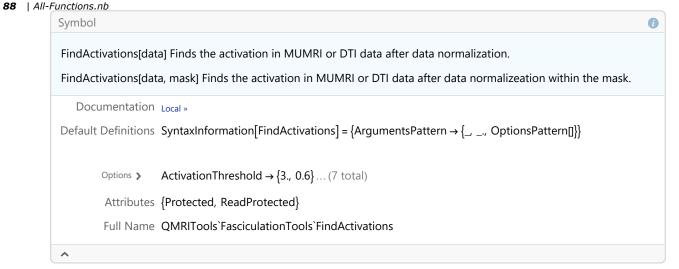


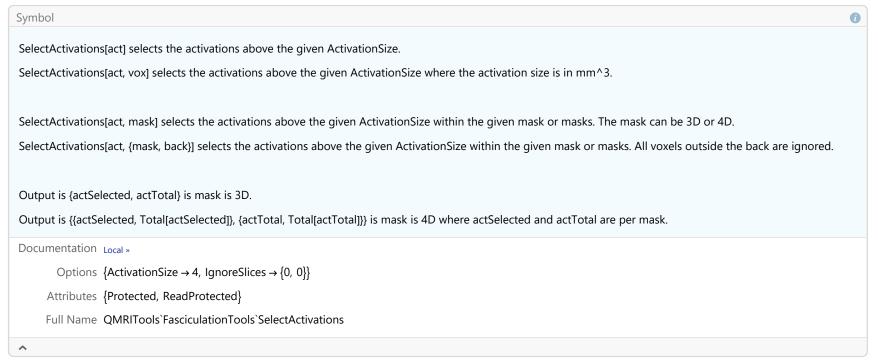
# **FasciculationTools**

### **Functions**

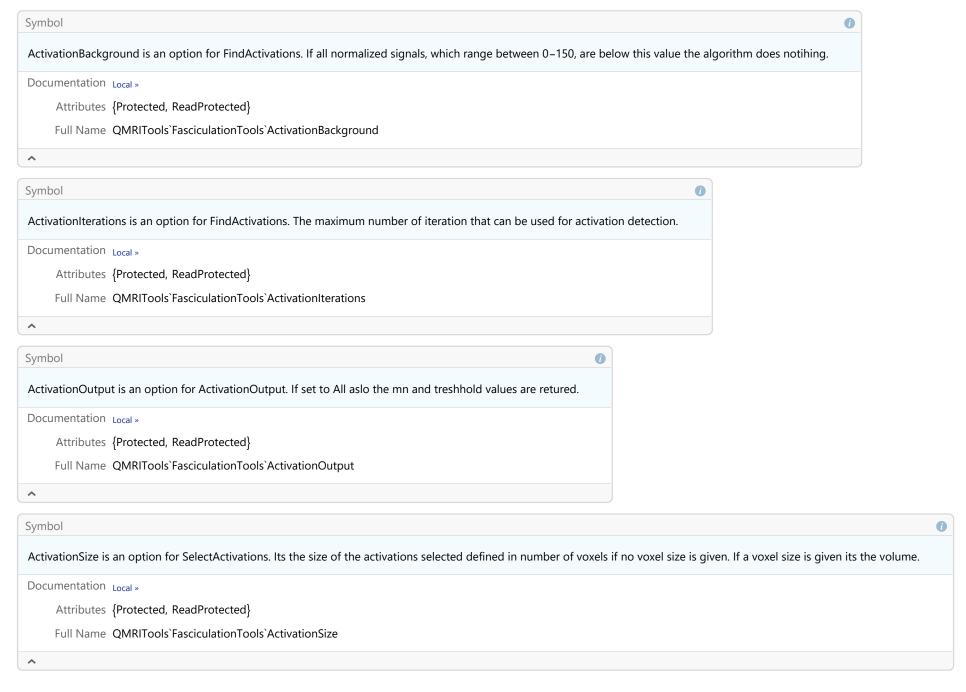


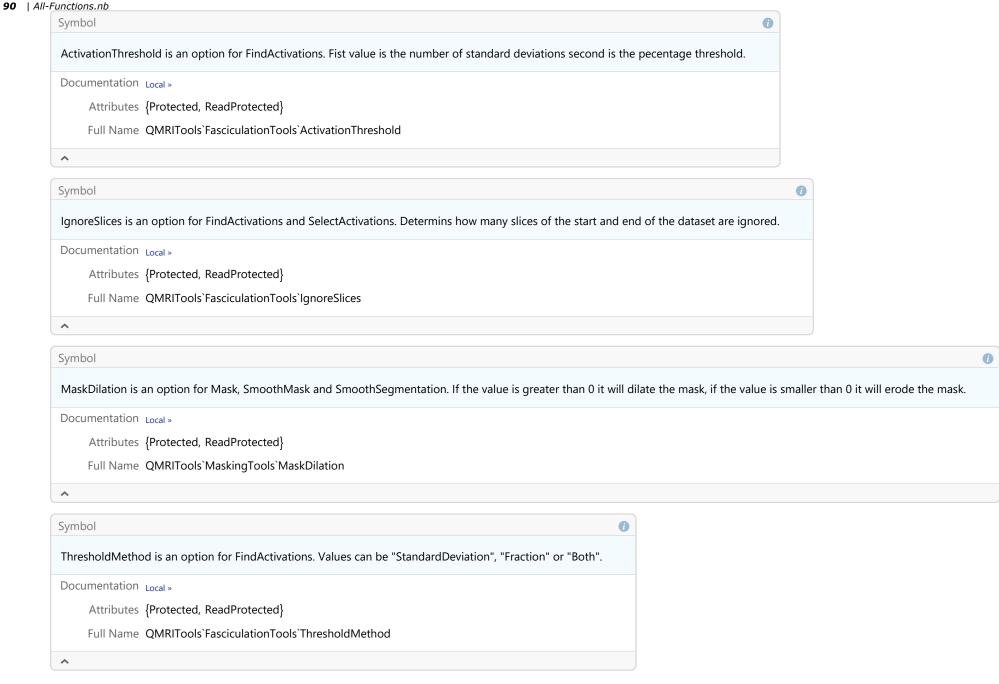






#### **Options**





# **GeneralTools**

#### **Functions**

```
ApplyCrop[data,crop] aplies the corpped region obtained form CropData to the data.

ApplyCrop[data,crop,{voxorig,voxnew}] aplies the corpped region obtained form CropData to the data.

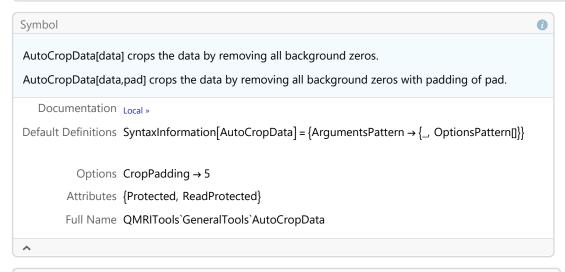
Documentation Local »

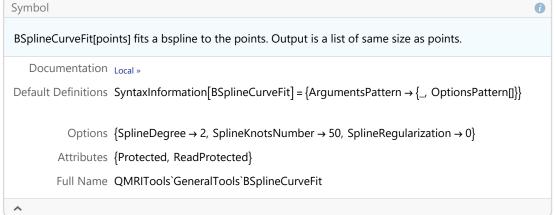
Default Definitions SyntaxInformation[ApplyCrop] = {ArgumentsPattern → {_, _, _, OptionsPattern[]}}

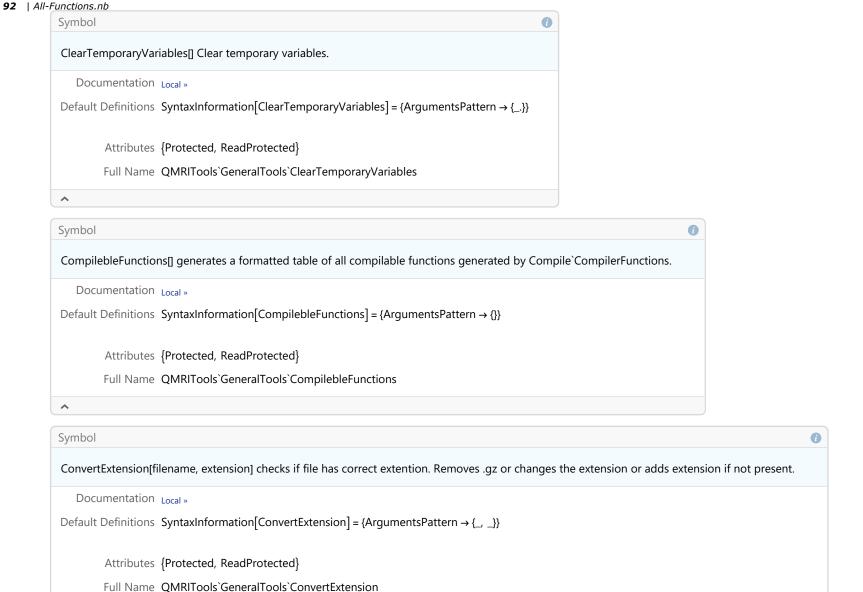
Options CropAlways → False

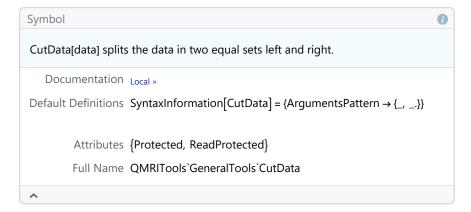
Attributes {Protected, ReadProtected}

Full Name QMRITools`GeneralTools`ApplyCrop
```

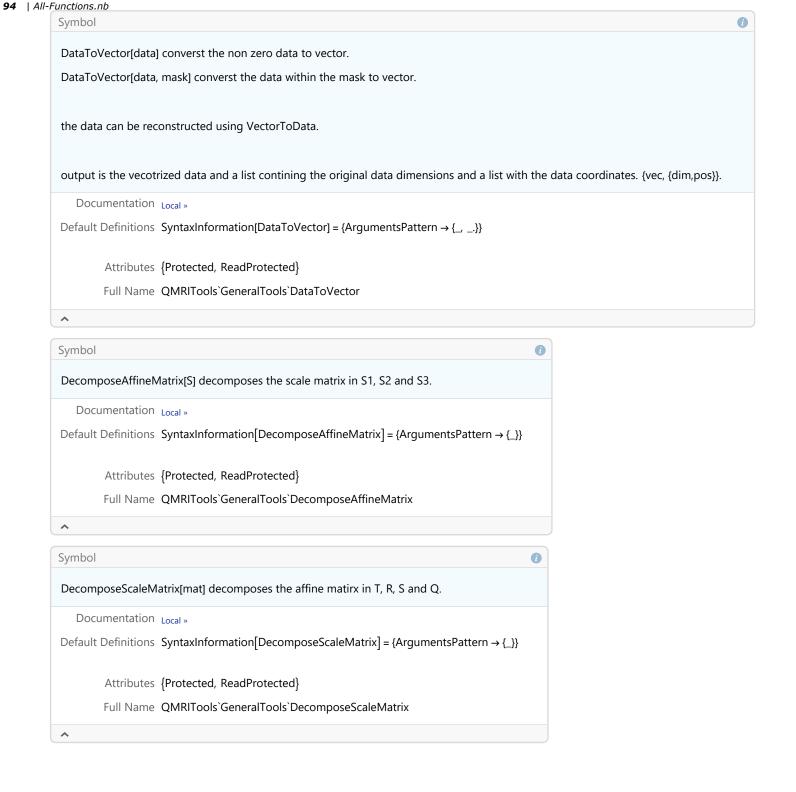


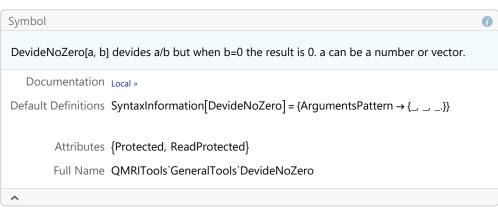


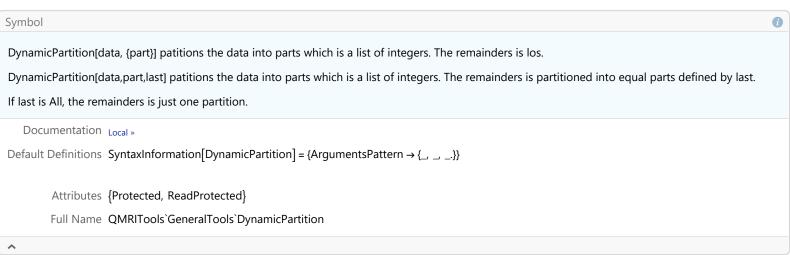


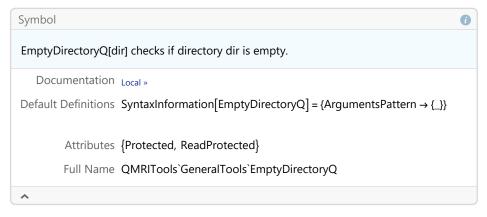


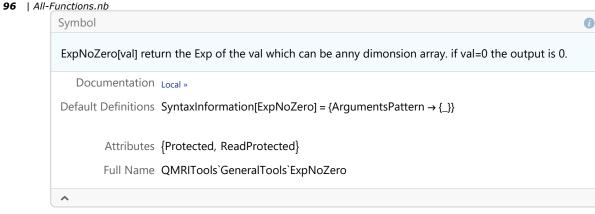
All-Functions.nb | 93

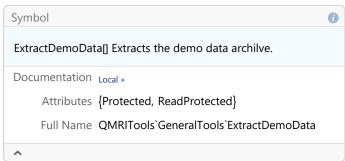


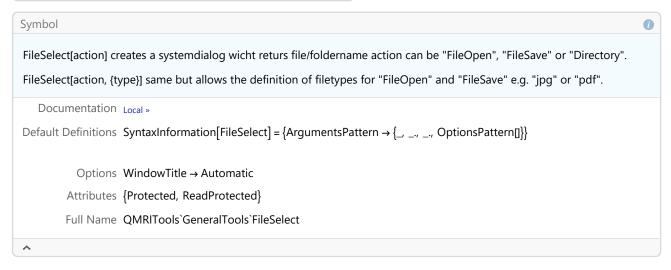


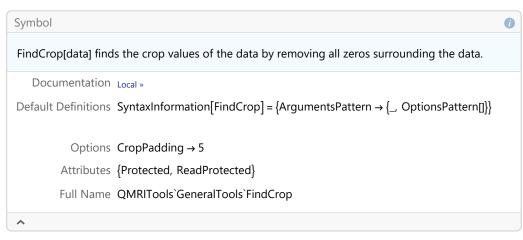


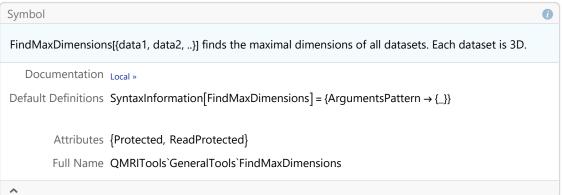


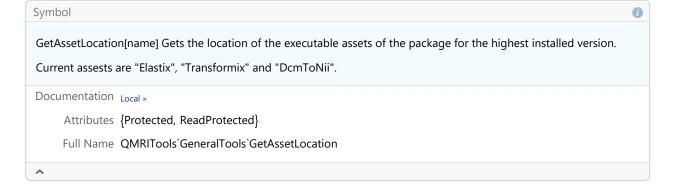


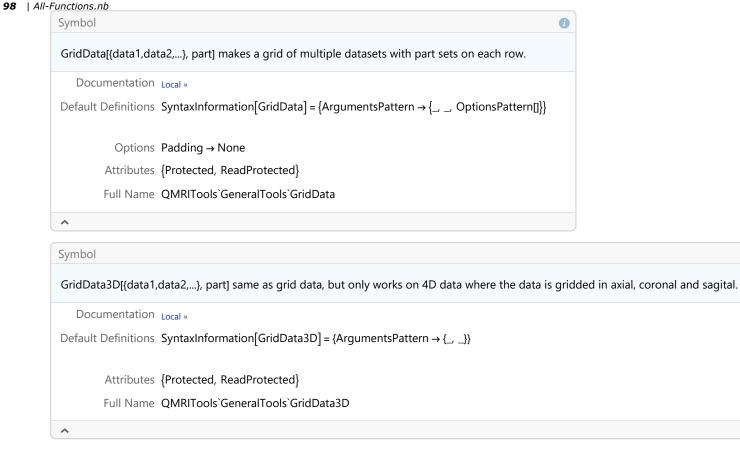


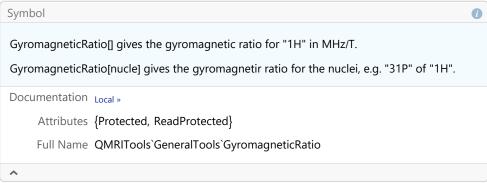


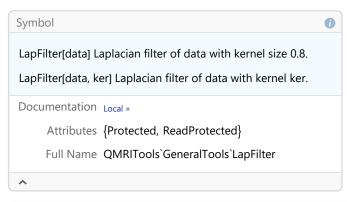


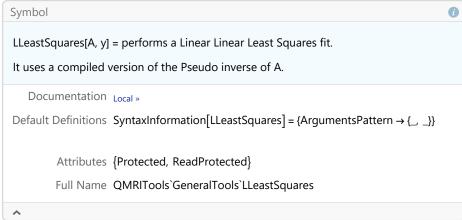


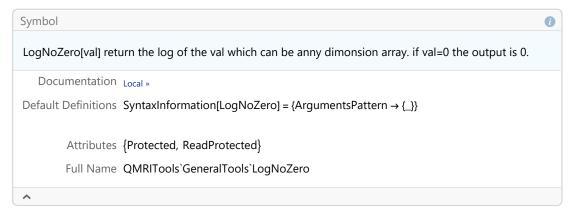


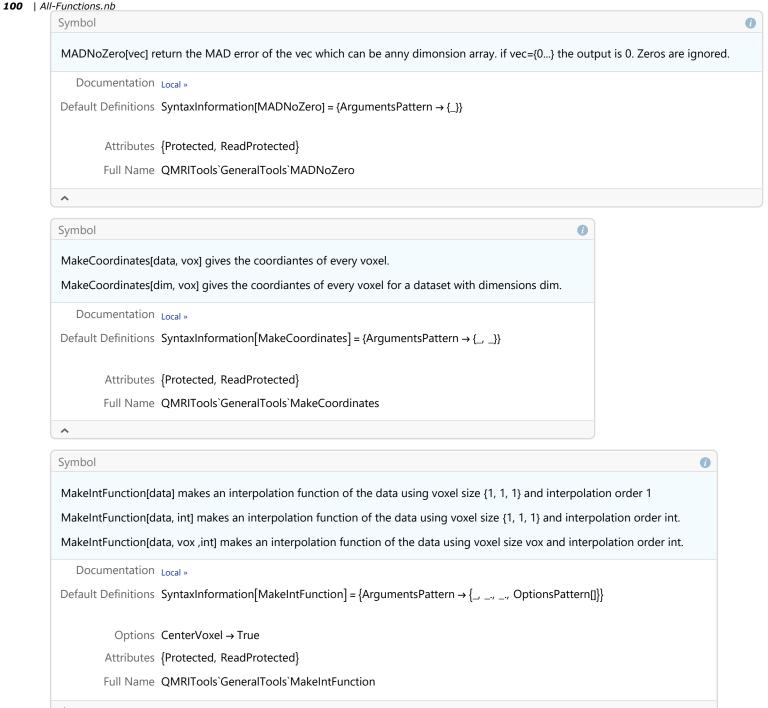


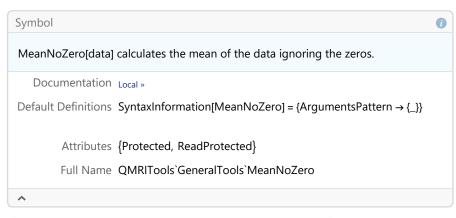


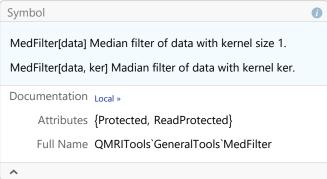


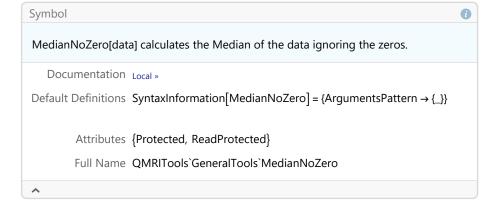


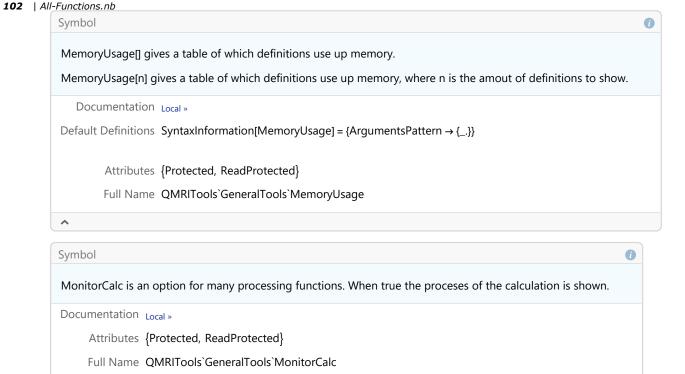


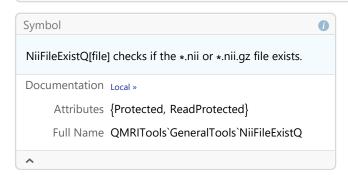




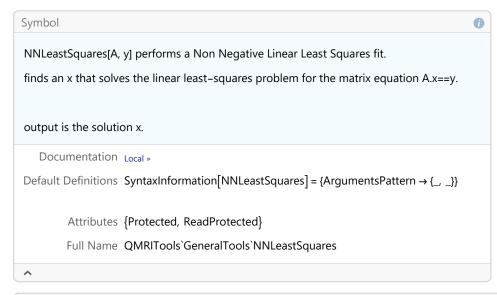


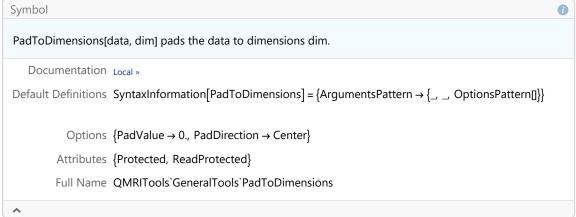


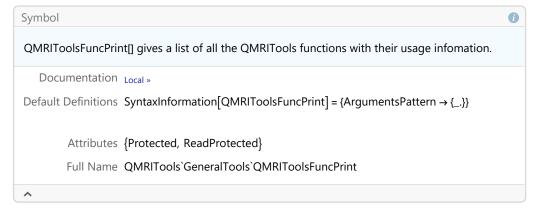


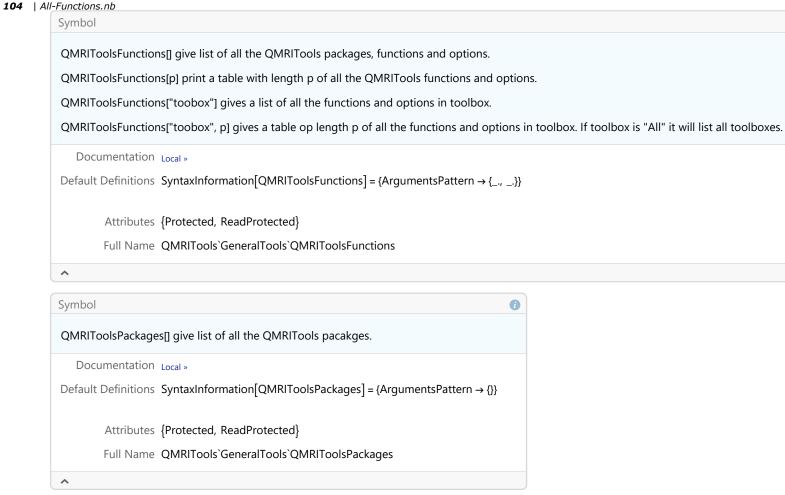


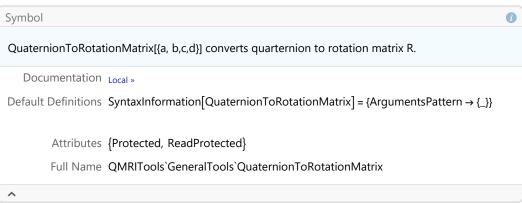
 $\wedge$ 

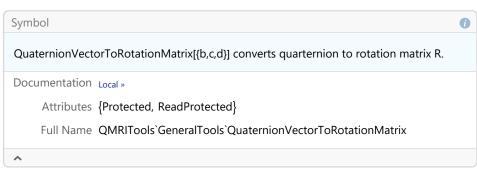


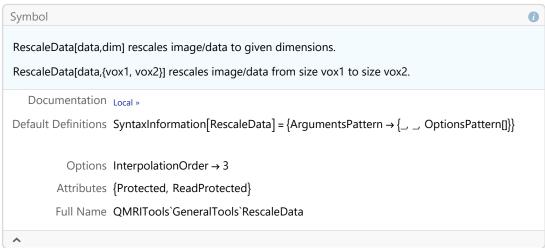


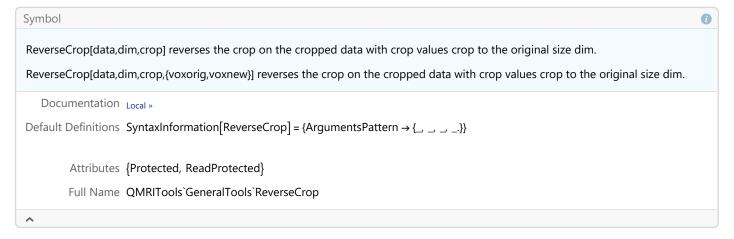


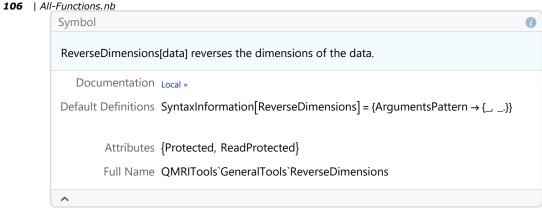


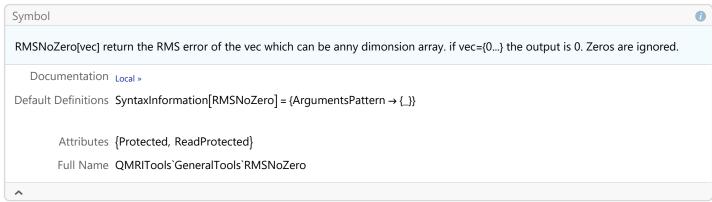


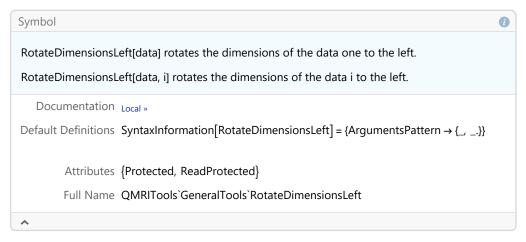


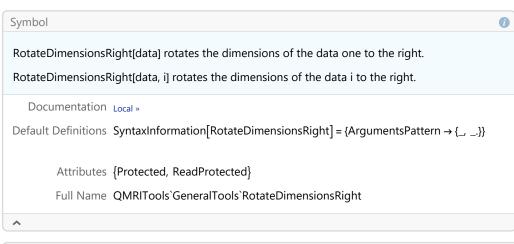


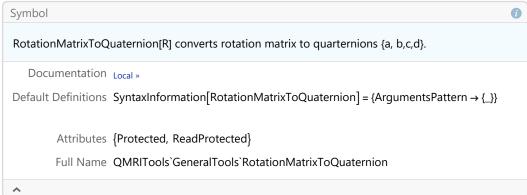


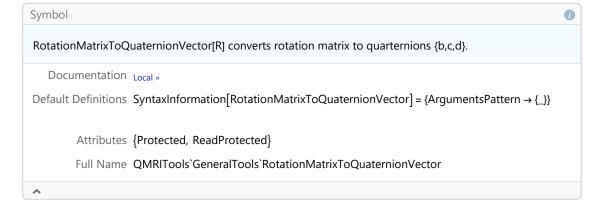


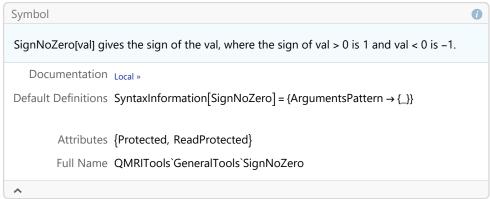


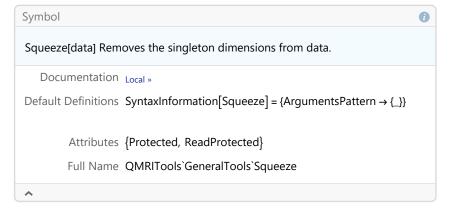


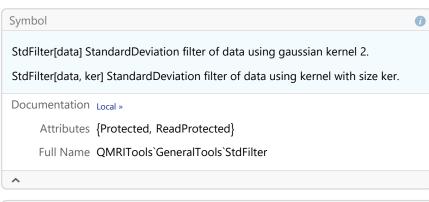


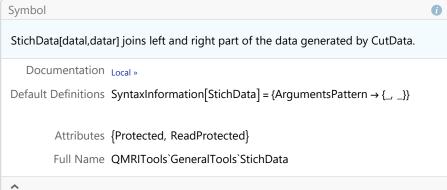


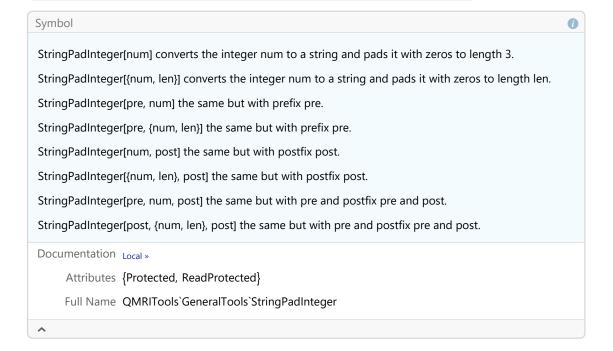


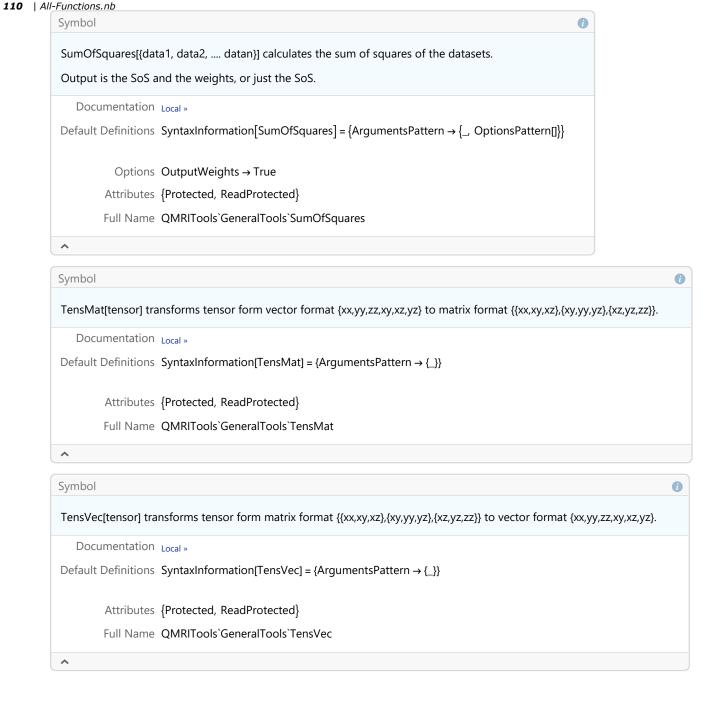




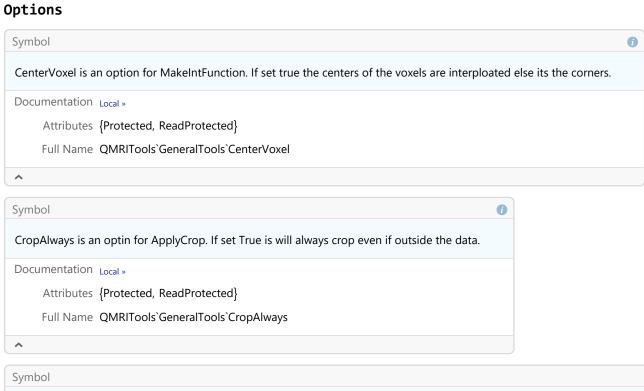


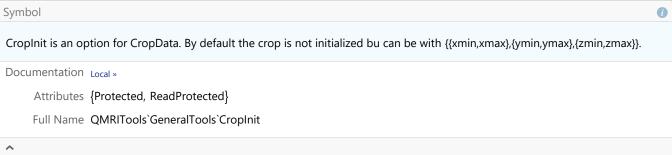


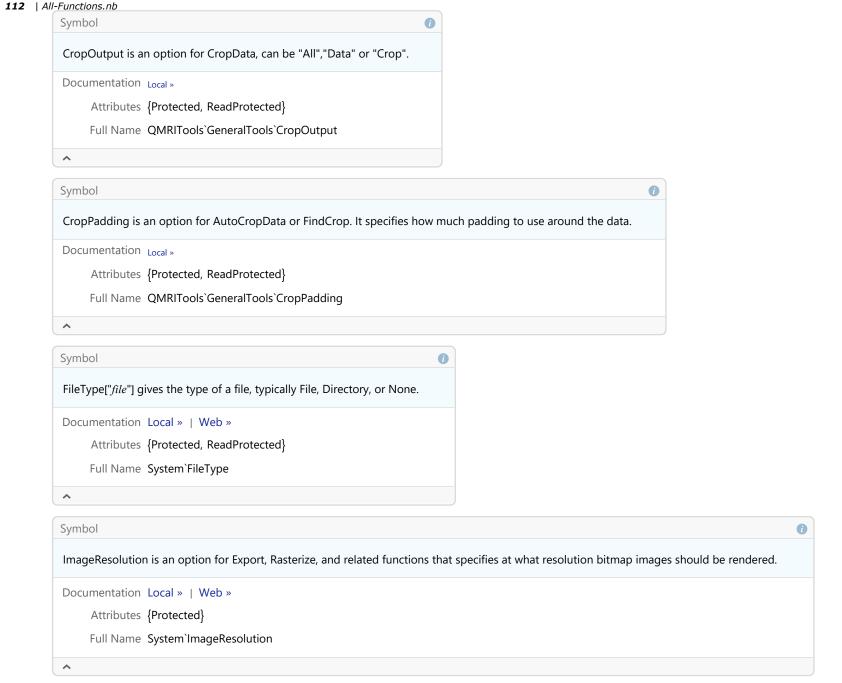




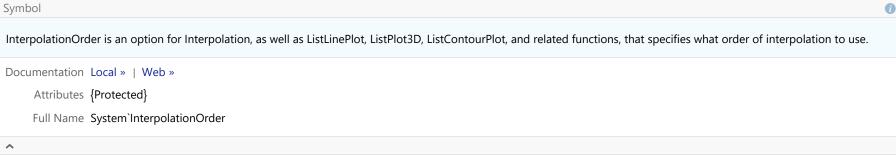
```
Symbol
VectorToData[vec, {dim,pos}] converts the vectroized data from DataToVector back to its original Dimensoins.
   Documentation Local »
Default Definitions SyntaxInformation[VectorToData] = {ArgumentsPattern → {_, {_, _}}}}
        Attributes {Protected, ReadProtected}
        Full Name QMRITools`GeneralTools`VectorToData
^
```

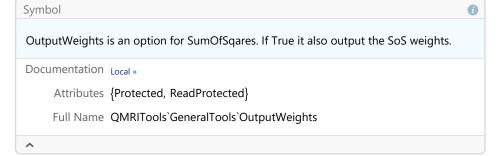


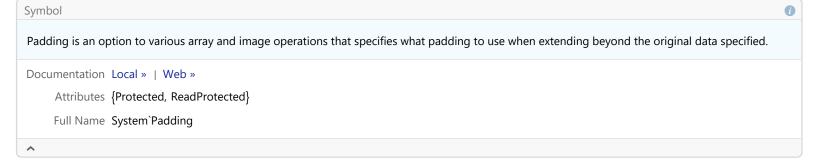


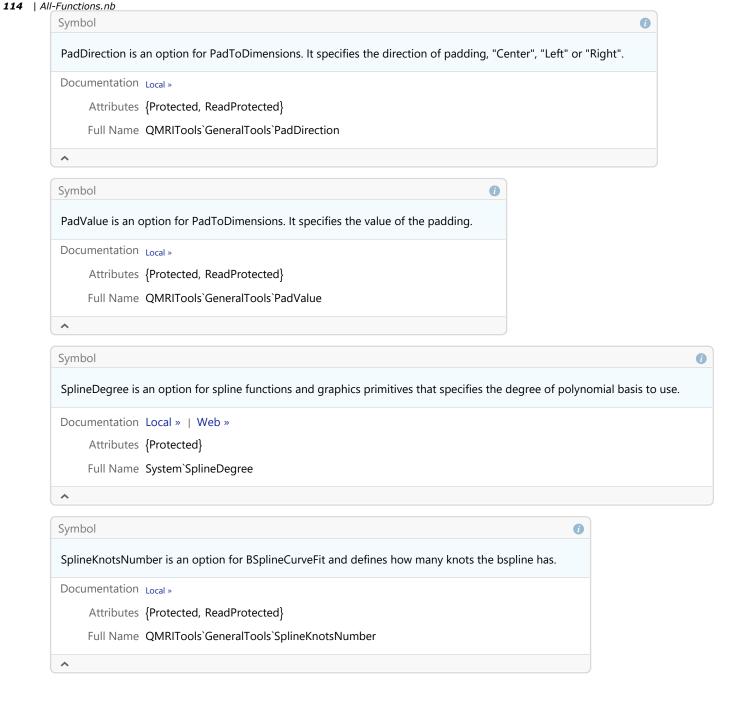












```
Symbol

SplineRegularization is an option for BSplineCurveFit and defines the amount of regularization for the linear fit.

Documentation Local »

Attributes {Protected, ReadProtected}

Full Name QMRITools`GeneralTools`SplineRegularization

Symbol

WindowTitle is an option that specifies the title to give for a window.

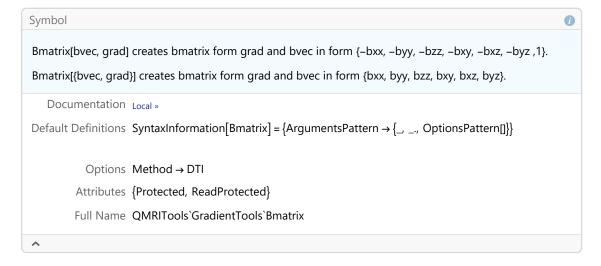
Documentation Local » | Web »

Attributes {Protected}

Full Name System`WindowTitle
```

# **GradientTools**

### **Functions**



BmatrixCalc["folder", grads] calculates the true bmatrix from the exported sequence parameters from the phillips scanner that are stored in "folder" for each of the gradient directions grads.

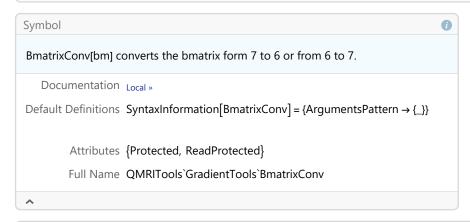
Documentation Local >

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

Options > UseGrad → {1, 1, {1, 1}, 1, 1}...(8 total)

Attributes {Protected, ReadProtected}

Full Name QMRITools'GradientTools'BmatrixCalc



^

BmatrixInv[bm] generates a bvecotr and gradiens directions form a given bmatrx.

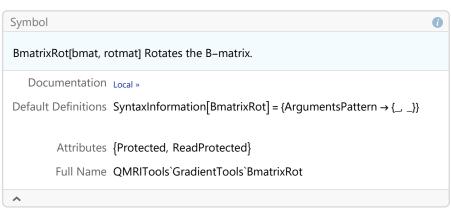
BmatrixInv[bm, bvi] generates a bvecotr and gradiens directions form a given bmatrx using the given bvalues bvi.

Documentation Local »

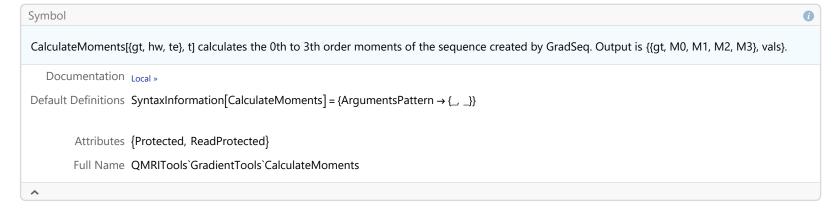
Default Definitions SyntaxInformation[BmatrixInv] = {ArgumentsPattern → {\_\_\_\_}}}

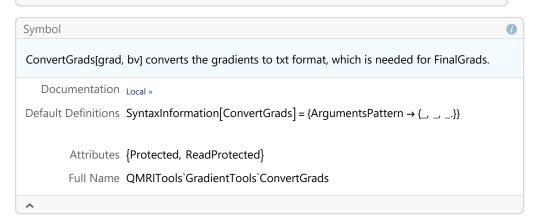
Attributes {Protected, ReadProtected}

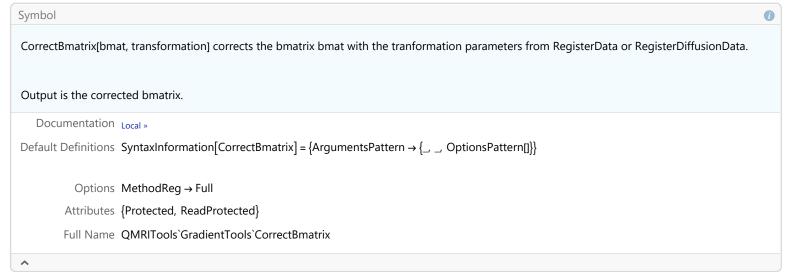
Full Name QMRITools`GradientTools`BmatrixInv











CorrectGradients[grad, transformation] corrects the gradient directions grad with the tranformation parameters from RegisterData or RegisterDiffusionData.

Output is the corrected gradient vector.

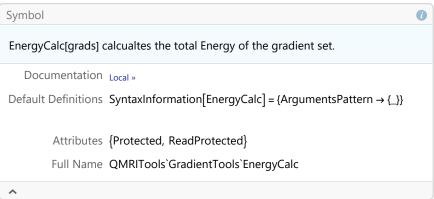
Documentation Local »

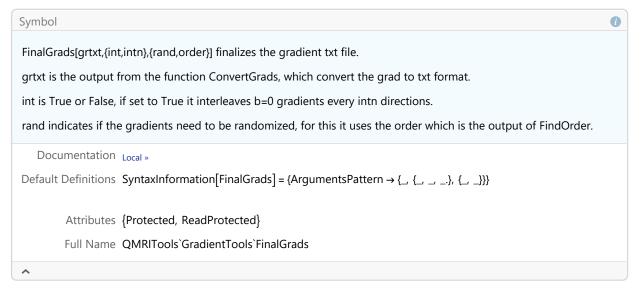
Default Definitions SyntaxInformation[CorrectGradients] = {ArgumentsPattern → {\_\_\_\_, OptionsPattern[]}}

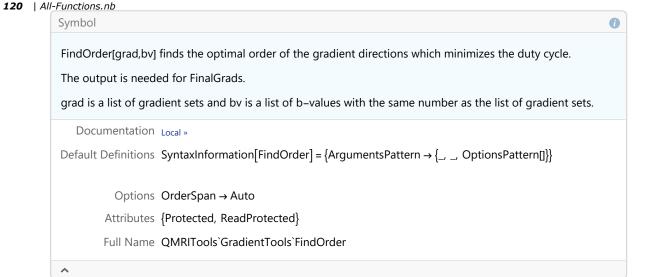
Options MethodReg → Rotation

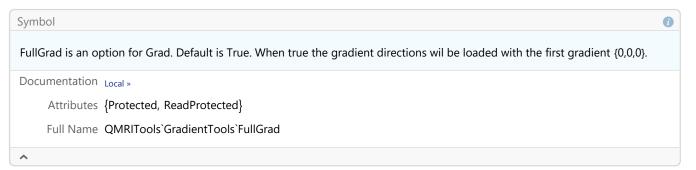
Attributes {Protected, ReadProtected}

Full Name QMRITools'GradientTools'CorrectGradients









GenerateGradients[numb] optimizes a set with numb gradients, numb mus be an integer.

GenerateGradients[{numb, fixed}] optimizes a set with numb gradients, numb must ba an integer and fixed a list of 3D coordiantes e.g. {{0,0,1},{0,1,0}}. The fixed gradients will not be moved.

0

GenerateGradients[{numb1, numb2 ...}, alpha] optimizes a multi shel gradient set with numb gradients per shel. If alpha is set to 0.5 equal importance is given

to the optimal distribution of each shell en the enitre set. if alpha is 0 only the sub shels will be optimized, if alpha is set to 1 only the global set wil be optimized.

GenerateGradients[] is based on DOI: 10.1002/mrm.26259 and 10.1002/(SICI)1522-2594(199909)42:3<515::AID-MRM14>3.0.CO;2-Q.

Documentation Local »

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

Options  $\Rightarrow$  Steps  $\Rightarrow$  1000 ... (6 total)

Attributes {Protected, ReadProtected}

Full Name QMRITools`GradientTools`GenerateGradients

^

#### Symbol

GenerateGradientsGUI[] runs the GenerateGradients function in GUI with output for the philips system.

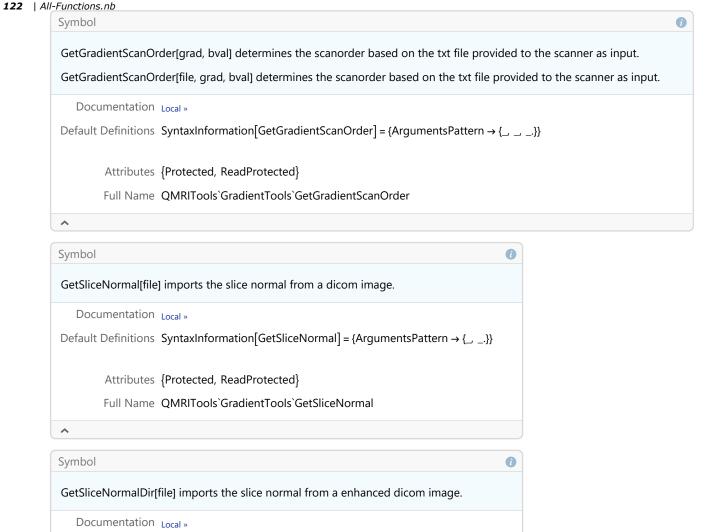
GenerateGradientsGUI[] is based on DOI: 10.1002/mrm.26259 and 10.1002/(SICI)1522-2594(199909)42:3<515::AID-MRM14>3.0.CO;2-Q.

Documentation Local »

Attributes {Protected, ReadProtected}

Full Name QMRITools`GradientTools`GenerateGradientsGUI

^

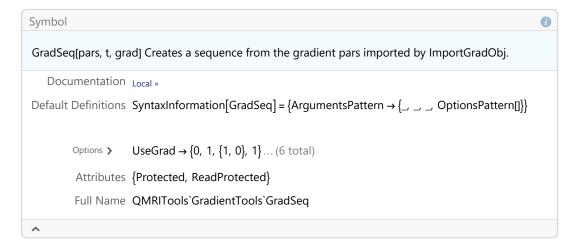


 $\label{eq:definitions} Default \ Definitions \ \ SyntaxInformation[GetSliceNormalDir] = \{ArgumentsPattern \rightarrow \{\_\}\}$ 

Full Name QMRITools`GradientTools`GetSliceNormalDir

Attributes {Protected, ReadProtected}

All-Functions.nb | 123



 $\wedge$ 

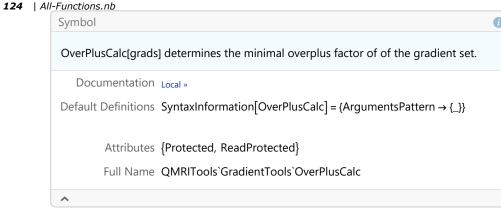
ImportGradObj[folder] Imports the gradient par files exported from the philips scanner.

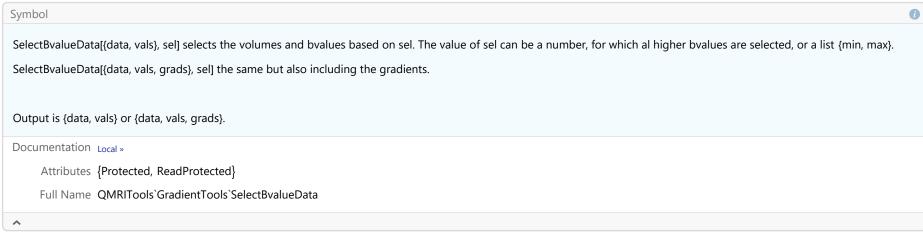
Documentation Local »

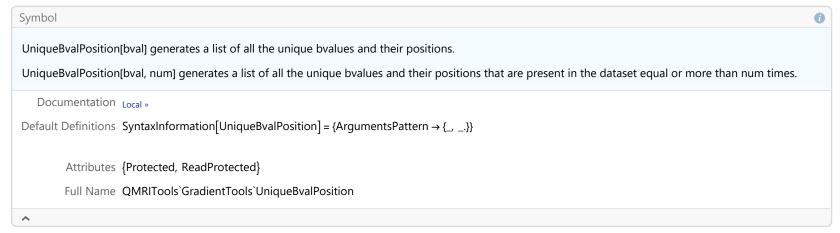
Default Definitions SyntaxInformation[ImportGradObj] = {ArgumentsPattern → {\_}}}

Attributes {Protected, ReadProtected}

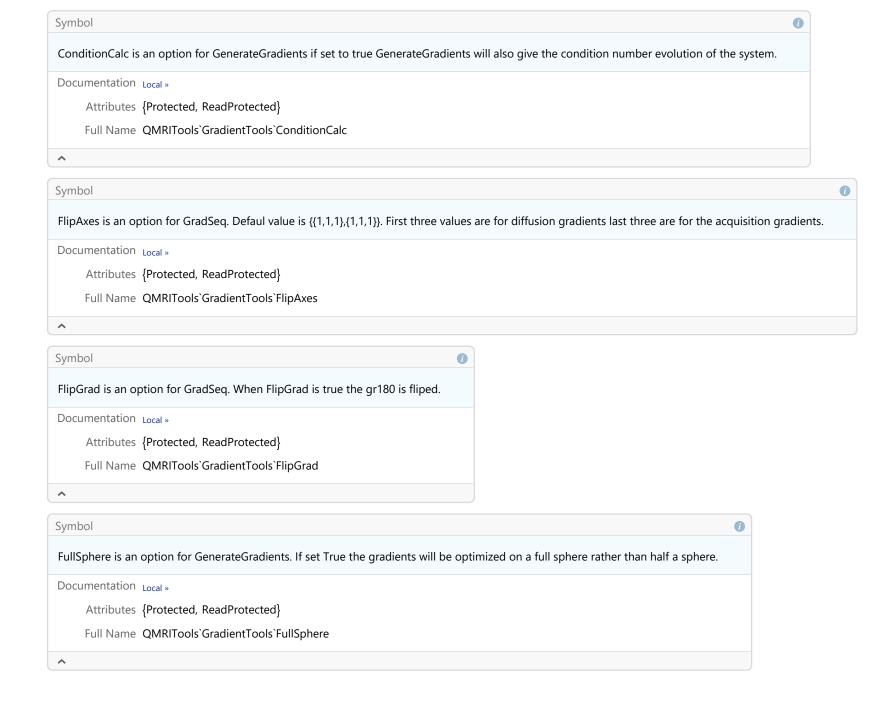
Full Name QMRITools`GradientTools`ImportGradObj





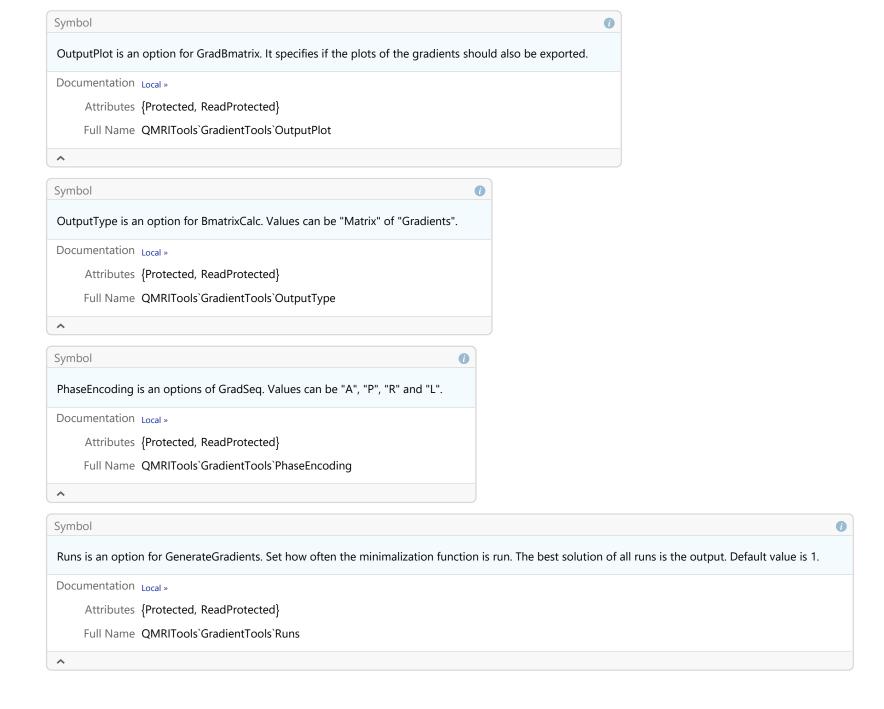


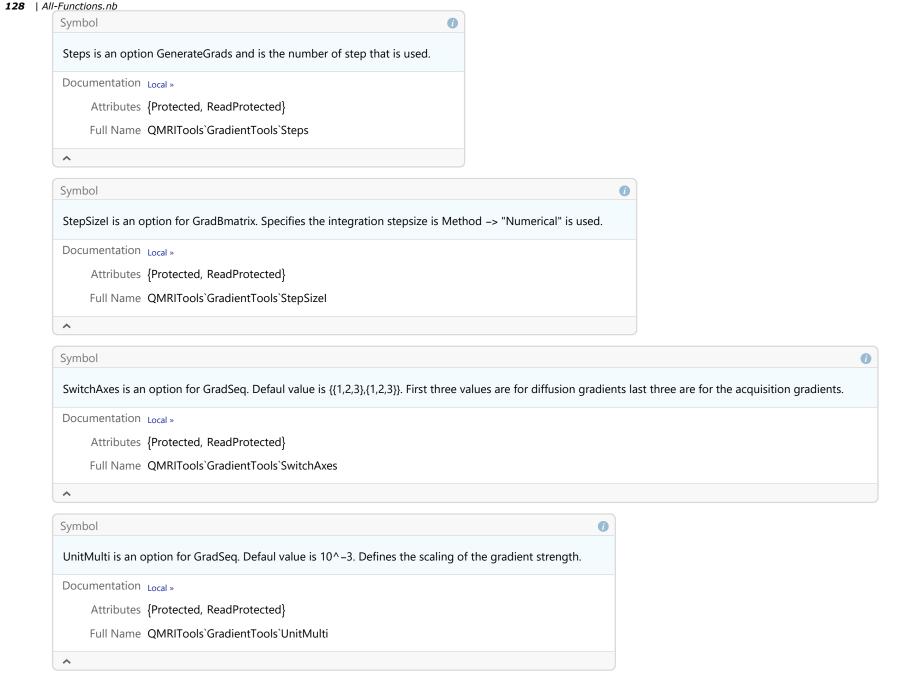
# **Options**

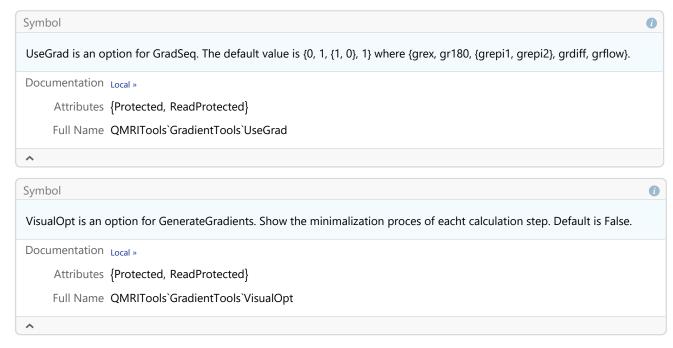


Full Name QMRITools`GradientTools`OrderSpan

^

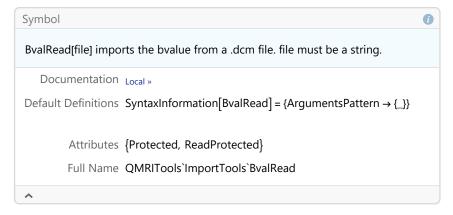


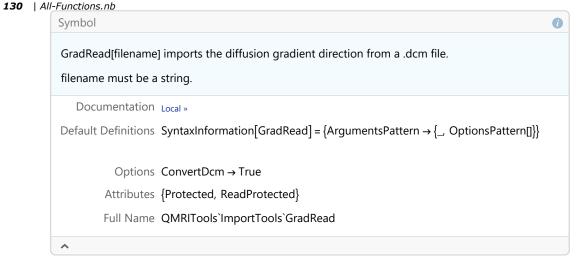


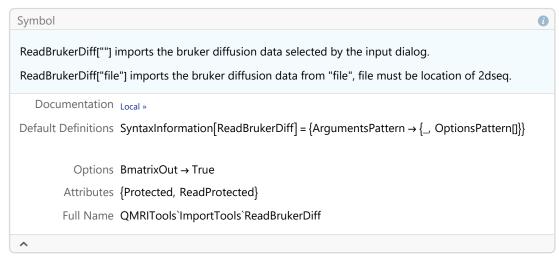


# **ImportTools**

## **Functions**









```
ReadDicom[folder] imports all dicom files from the given folder.

ReadDicom[file1, file2,...] imports all the given filenames.

ReadDicom[folder, {file1, file2,...}] imports all the given filenames from the given folder.

ReadDicom[folder, partsize] imports all dicom files from the given folder and partions them in given partsize.

ReadDicom[file1, file2, ...}, partsize] imports all the given filenames and partions them in given partsize.

ReadDicom[folder, {file1, file2, ...}, partsize] imports all the given filenames from the given folder and partions them in given partsize.

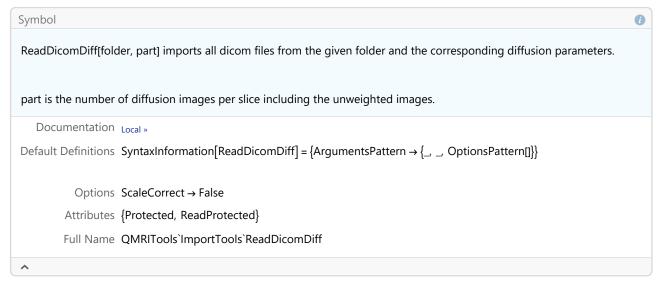
Documentation Local »

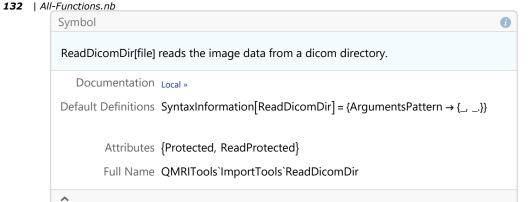
Default Definitions SyntaxInformation[ReadDicom] = {ArgumentsPattern → {__, __, __, OptionsPattern[]}}

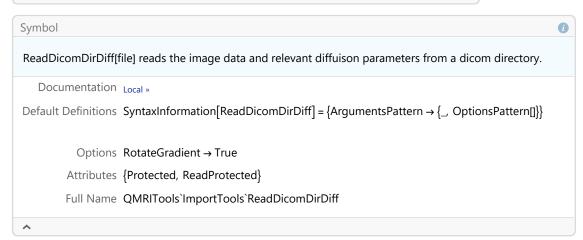
Options ScaleCorrect → False

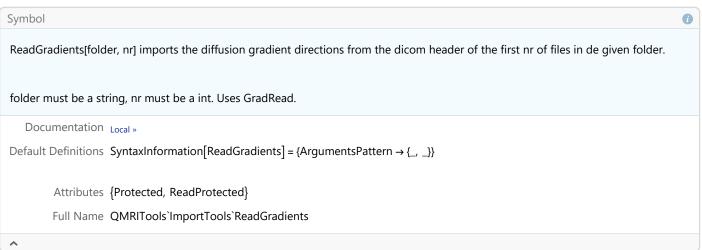
Attributes {Protected, ReadProtected}

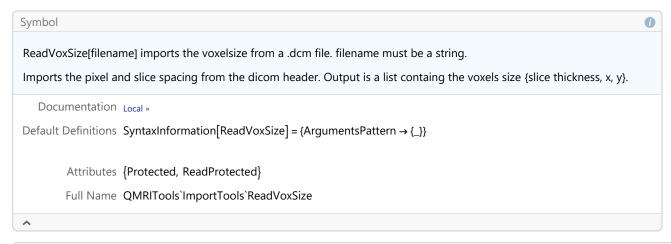
Full Name QMRITools'ImportTools'ReadDicom
```

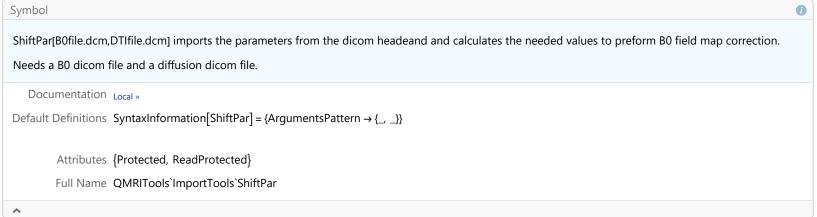




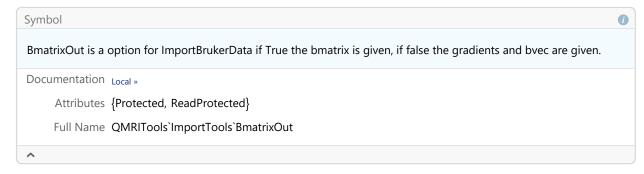


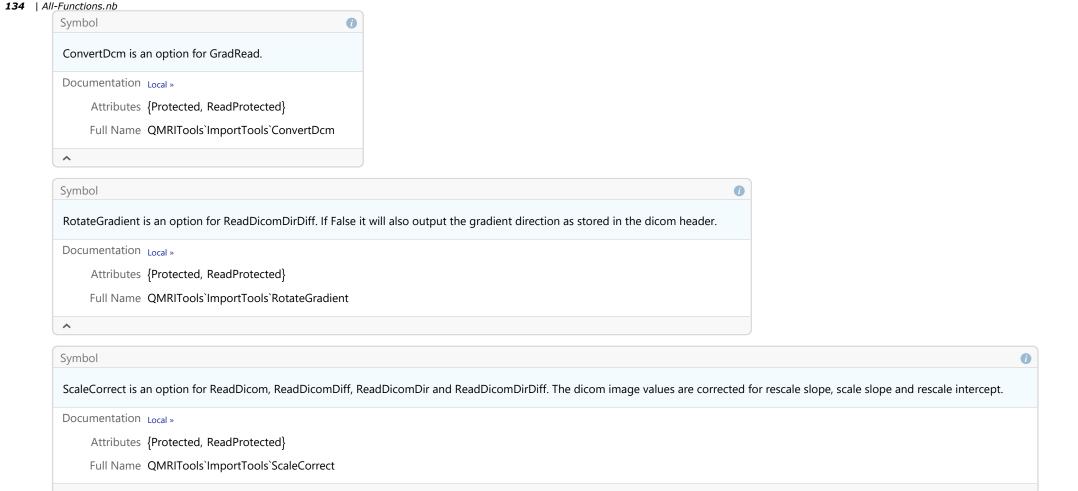






### **Options**



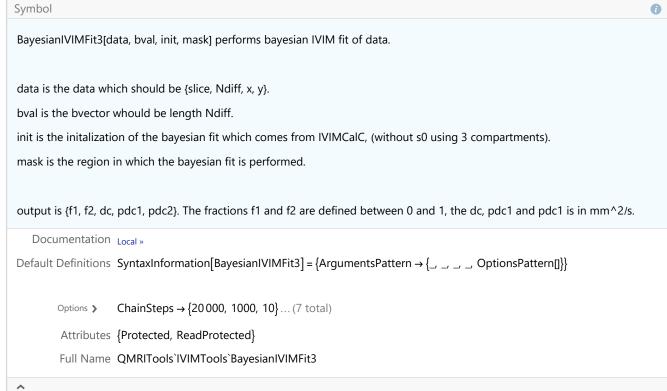


# **IVIMTools**

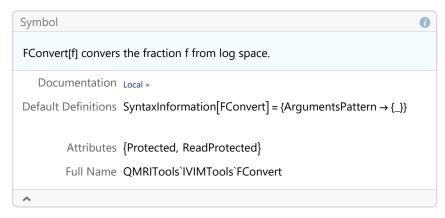
**Functions** 

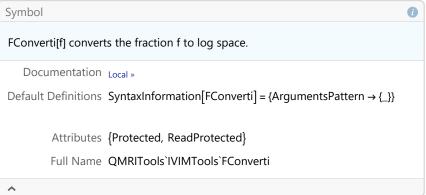
^

All-Functions.nb | 135









Symbol

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

After correction the signal fraction can be regarded as volume fraction.

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}, {t1t, t11}}.

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 → {\_, \_, \_, \_}}

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
^
```

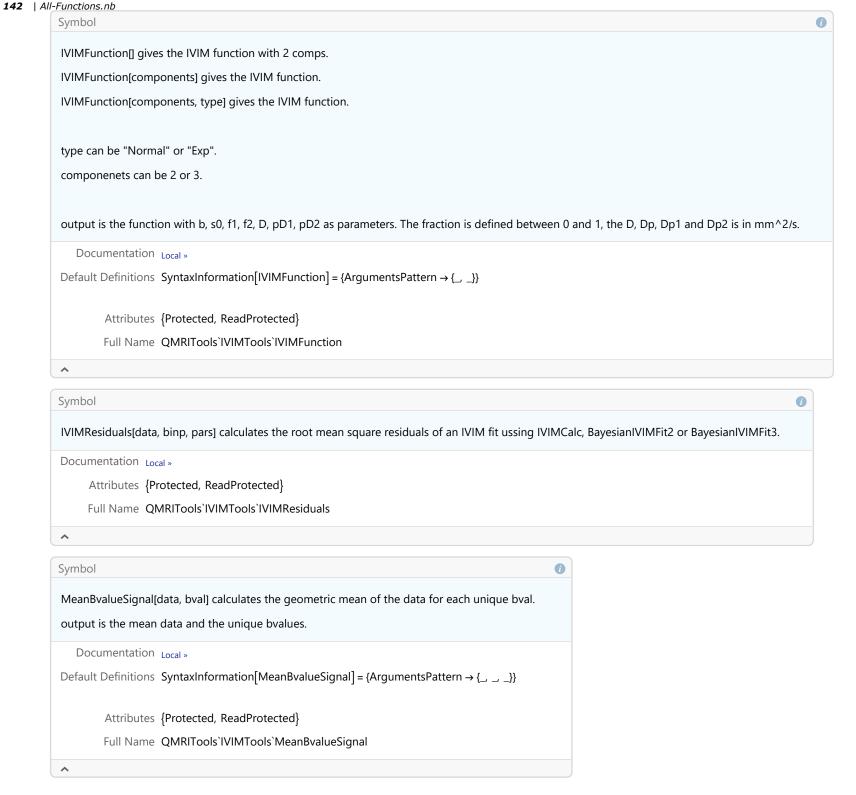
All-Functions.nb | 139

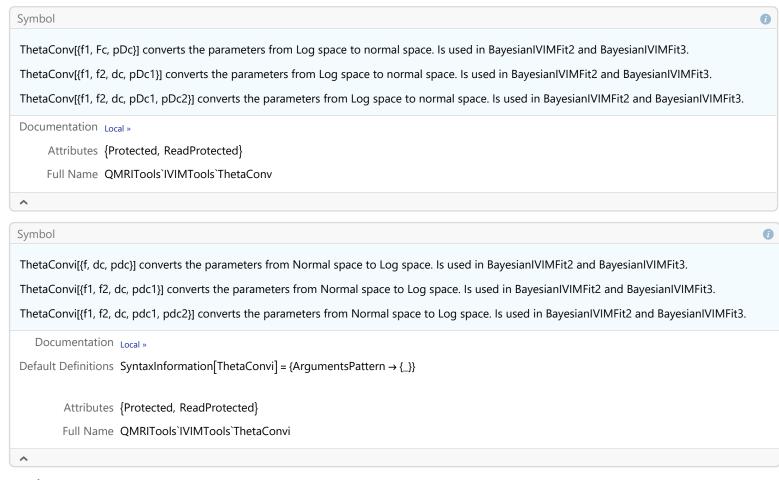
**140** | All-Functions.nb Symbol IVIMCalc[data, binp, init] calculates the IVIM fit. data should be 1D, 2D, 3D or 4D. binp should be full bmatrix which can be calculated from the byecs en byals using Bmatrix with the byalues in s/mm^2. init should are the initialization parameters for 2 components this 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] =  $\{ArgumentsPattern \rightarrow \{\_, \_, \_, OptionsPattern[]\}\}$ Method → Automatic ... (8 total) Options > Attributes {Protected, ReadProtected} Full Name QMRITools`IVIMTools`IVIMCalc

 $\wedge$ 

```
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 mm^2/s.
output is the corrected data.
   Documentation Local »
Default Definitions SyntaxInformation[IVIMCorrectData] = {ArgumentsPattern → {_, {_, _, _,}, _, OptionsPattern[]}}
           Options {FilterMaps \rightarrow True, FilterType \rightarrow Median, FilterSize \rightarrow 1}
         Attributes {Protected, ReadProtected}
         Full Name QMRITools`IVIMTools`IVIMCorrectData
^
```

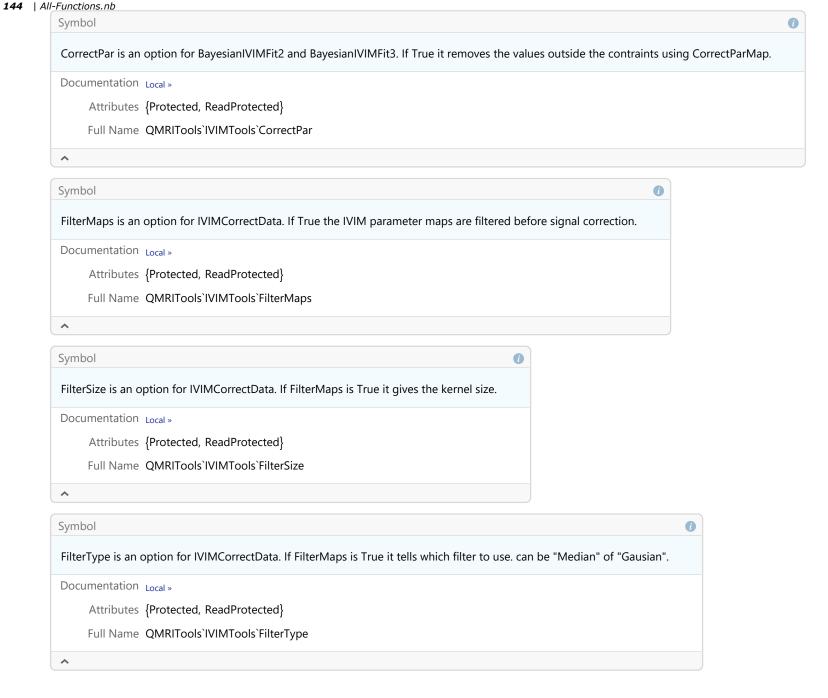
All-Functions.nb | 141

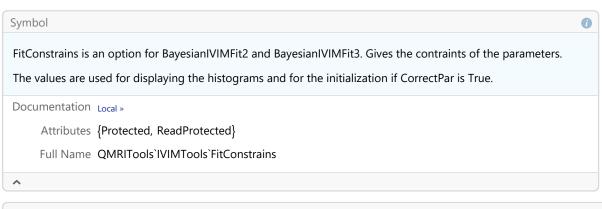


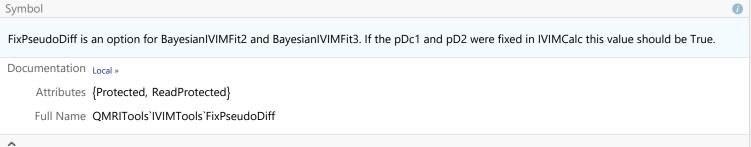


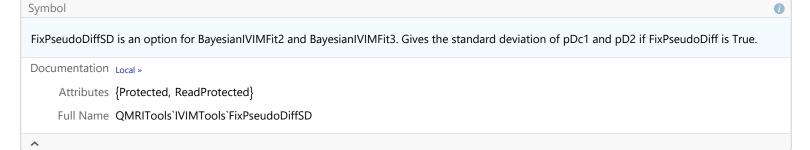
# **Options**

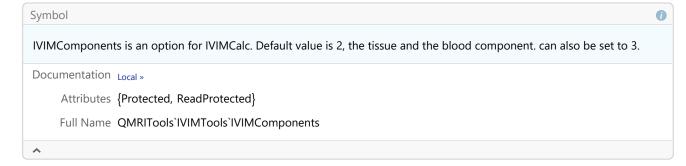


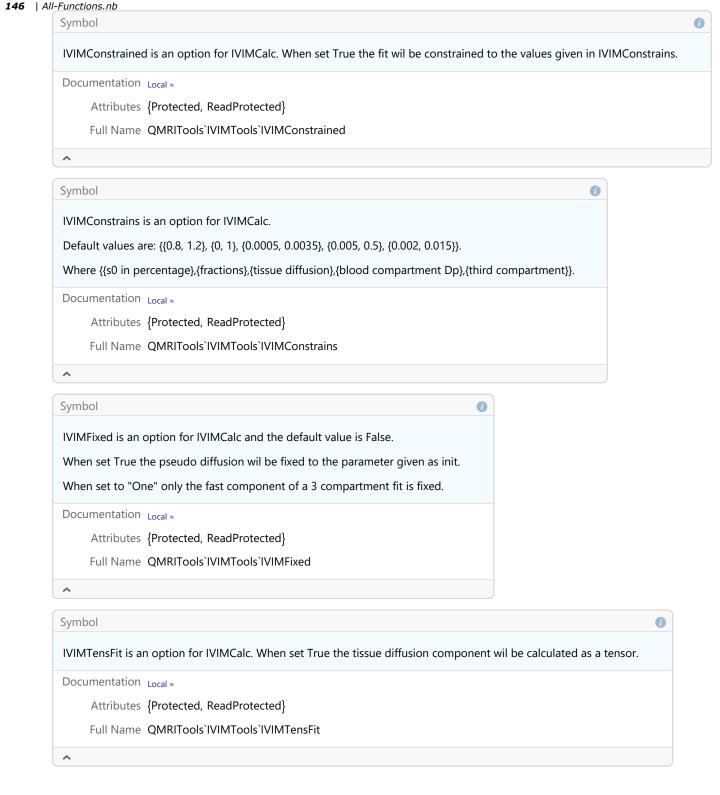


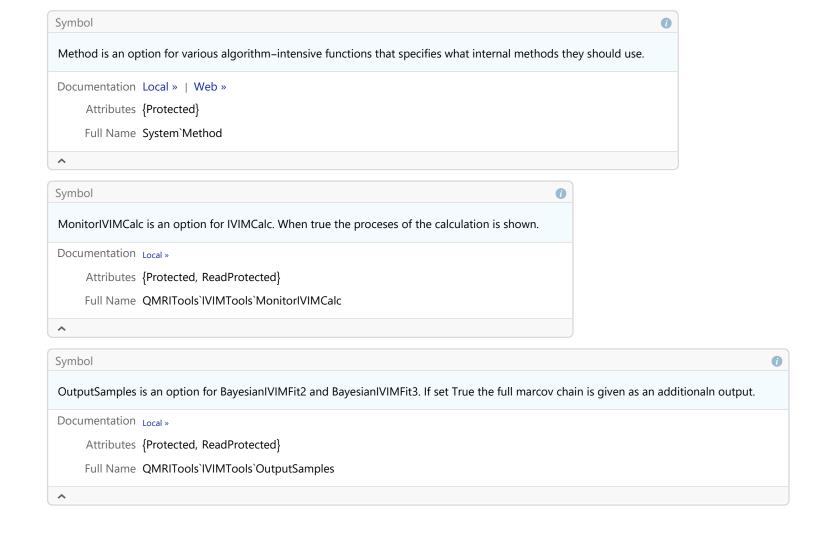












```
UpdateStep is an option for BayesianIVIMFit2 and BayesianIVIMFit3. It determines how often the parameters are updated. Is optimized during the first 500 burn steps.

Documentation Local »

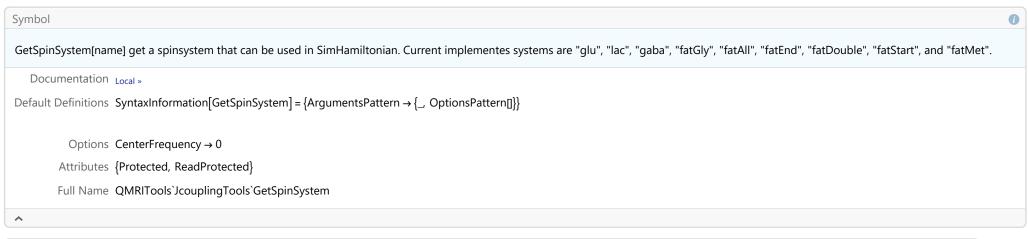
Attributes {Protected, ReadProtected}

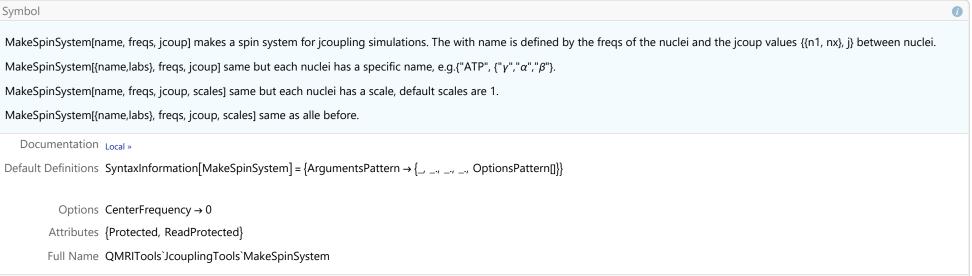
Full Name QMRITools`IVIMTools`UpdateStep
```

## **JcouplingTools**

#### **Functions**

 $\wedge$ 





^

# 

Full Name QMRITools`JcouplingTools`SequenceSteam

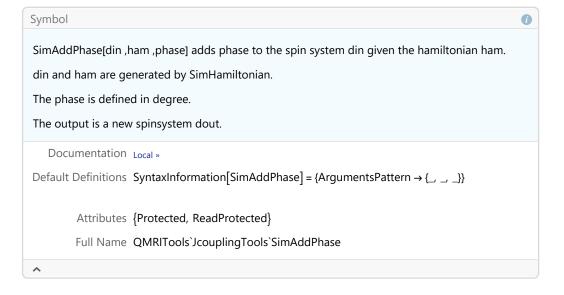
 $\wedge$ 

Documentation Local »

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

Attributes {Protected, ReadProtected}

Full Name QMRITools'JcouplingTools'SequenceTSE



```
SimEvolve[din,ham,t] evolves the spin system din given the hamiltonian ham over a time t. din and ham are generated by SimHamiltonian.

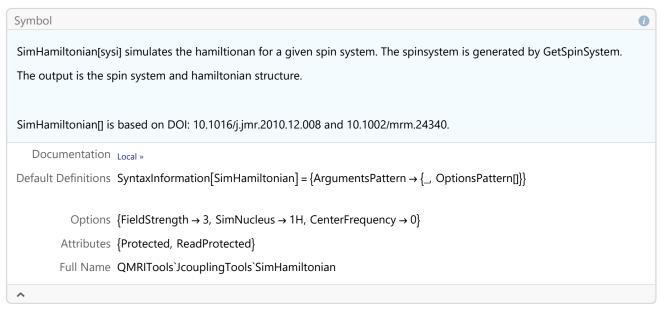
The output is a new spinsystem dout.

Documentation Local »

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

Attributes {Protected, ReadProtected}

Full Name QMRITools'JcouplingTools'SimEvolve
```



Symbol

SimReadout[din, ham] performs a readout of a spinsystem din with hamiltonian ham.

Output is {time,fids,ppm,spec,dout}, which are the free induction decay fids with its time, the spectrum spec with its ppm and the evolved spin system dout.

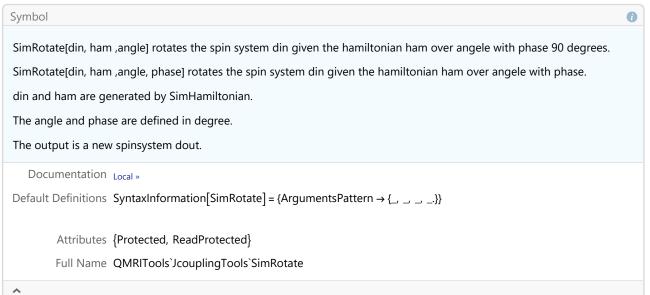
Documentation Local >>

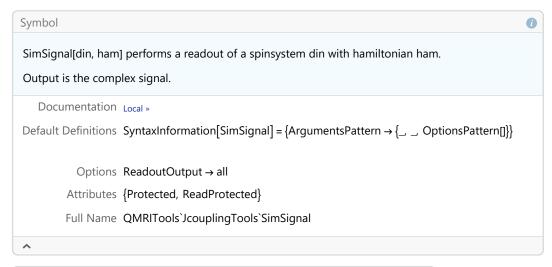
Default Definitions SyntaxInformation[SimReadout] = {ArgumentsPattern → { \_ \_ \_ \_ , OptionsPattern[]}}

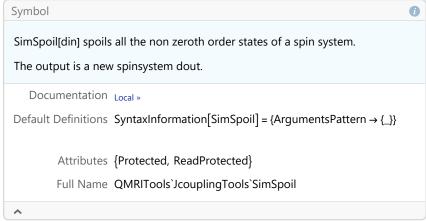
Options > ReadoutOutput → all ... (8 total)

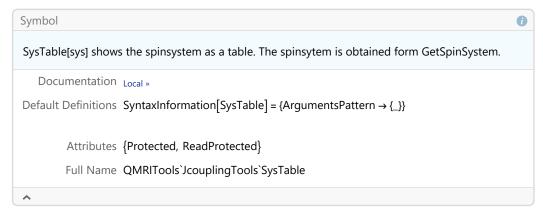
Attributes {Protected, ReadProtected}

Full Name QMRiTools'JcouplingTools'SimReadout

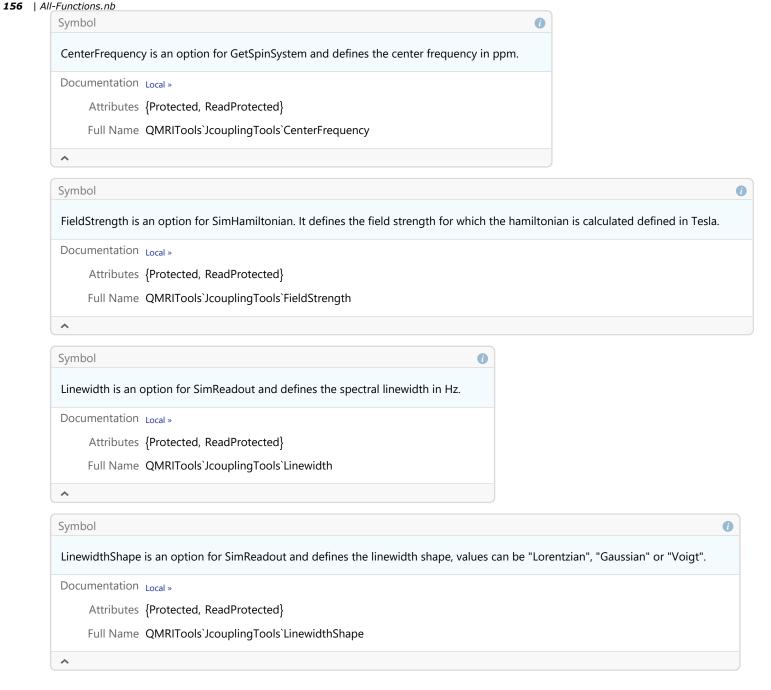


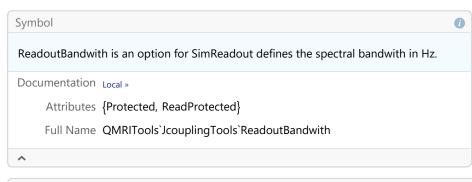


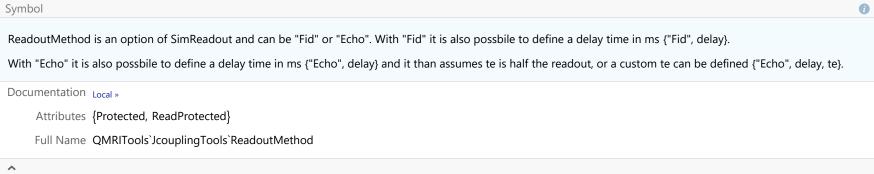




## **Options**







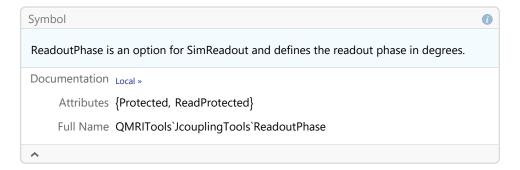
ReadoutOutput is an option for SimReadout and SimSignal and values can be "all" and "each".

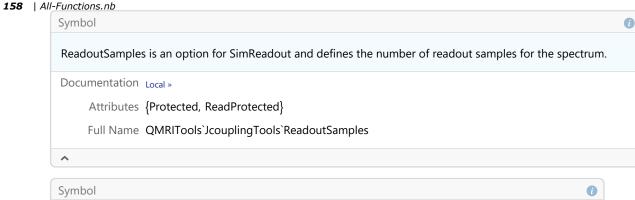
When set to "all" the total signal and signal is given, when set to "each" the signal or spectrum for each peak is given seperately.

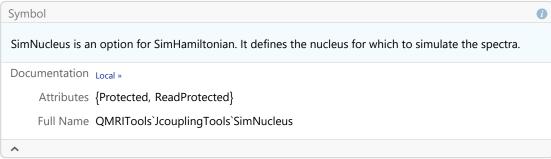
Documentation Local »

Attributes {Protected, ReadProtected}}

Full Name QMRITools'JcouplingTools'ReadoutOutput

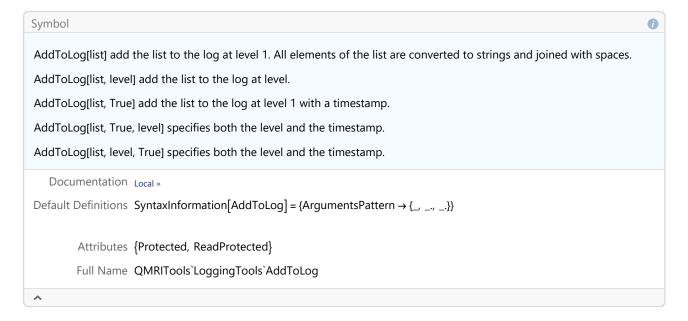


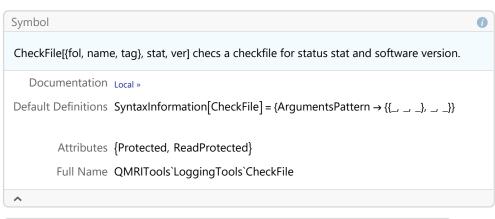


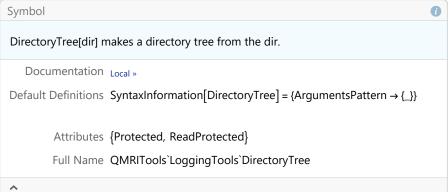


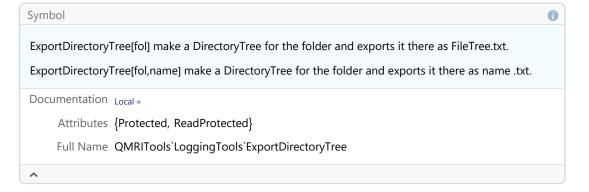
## LoggingTools

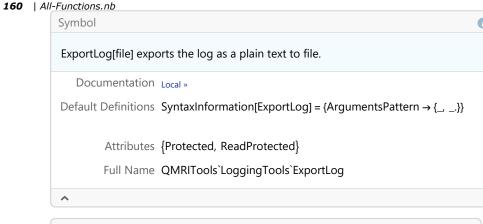
#### **Functions**

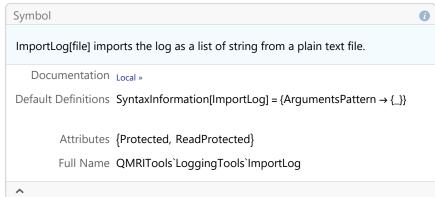


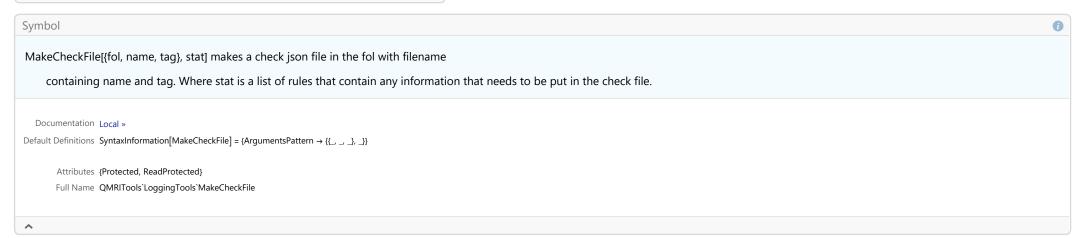


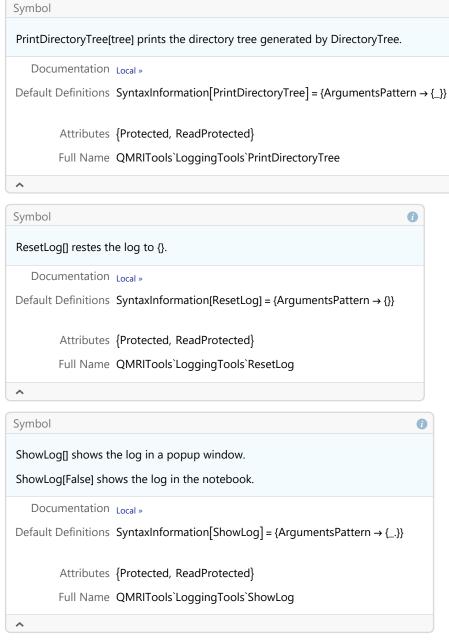








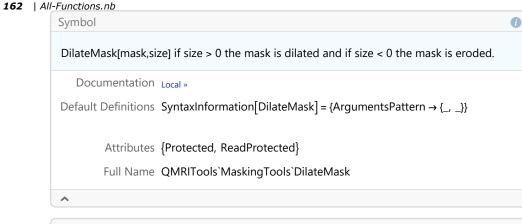


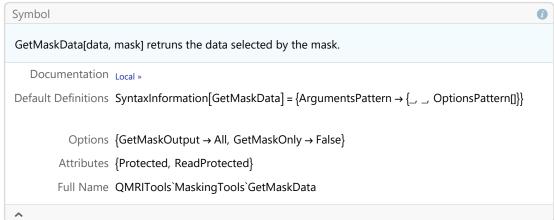


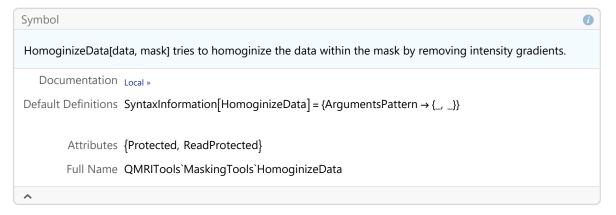
### **Options**

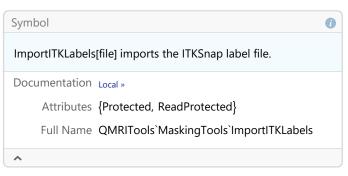
# MaskingTools

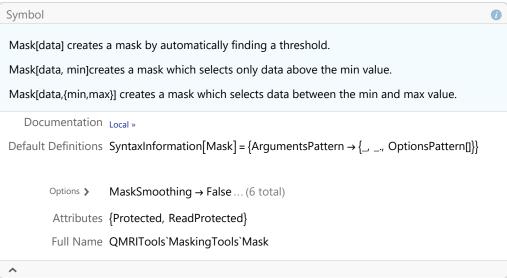
**Functions** 

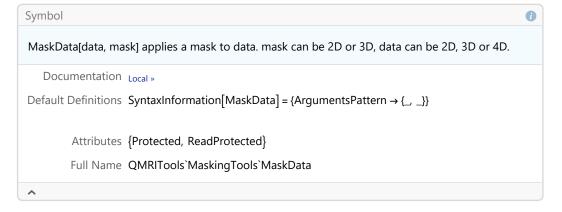


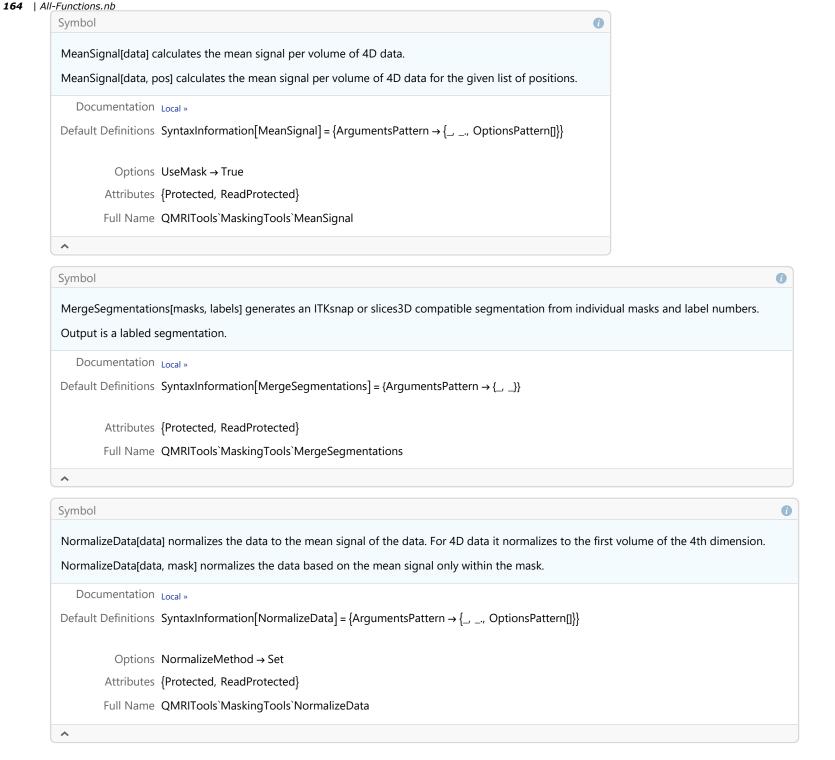


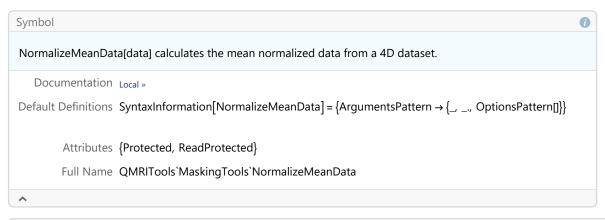


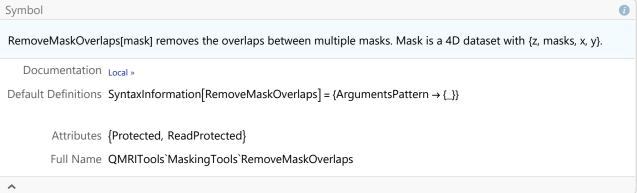


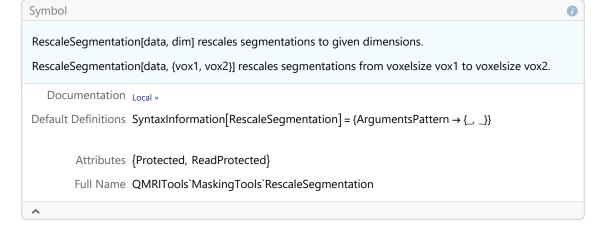


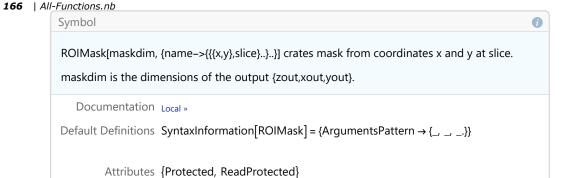




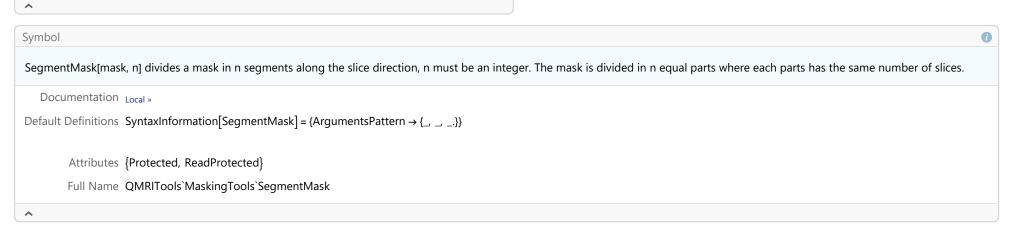


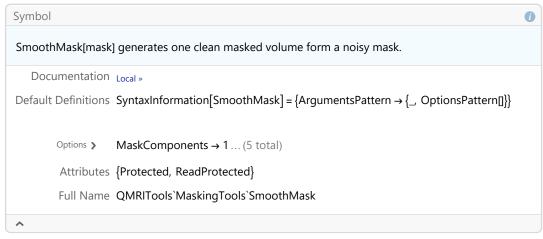






Full Name QMRITools`MaskingTools`ROIMask





Symbol

SmoothSegmentation[masks] smooths segmentations and removes the overlaps between multiple segmentations.

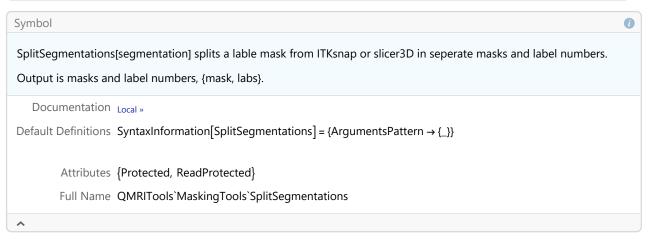
Documentation Local »

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

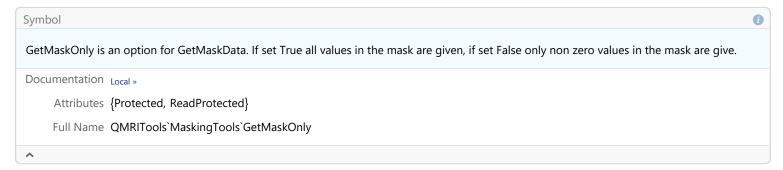
Options > MaskComponents → 2 ... (5 total)

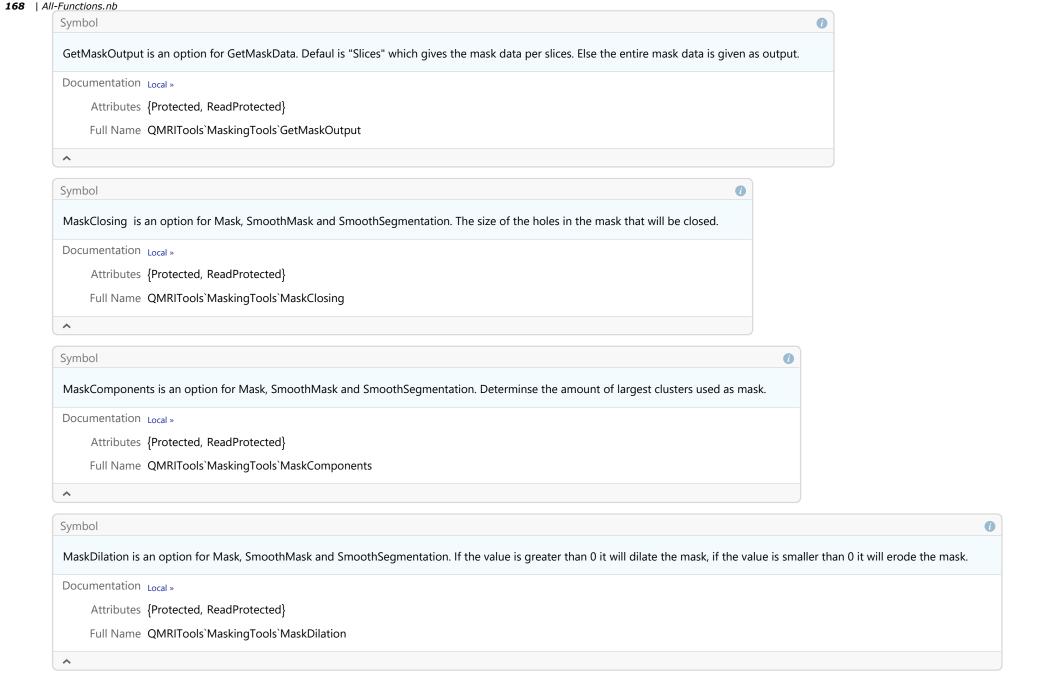
Attributes {Protected, ReadProtected}

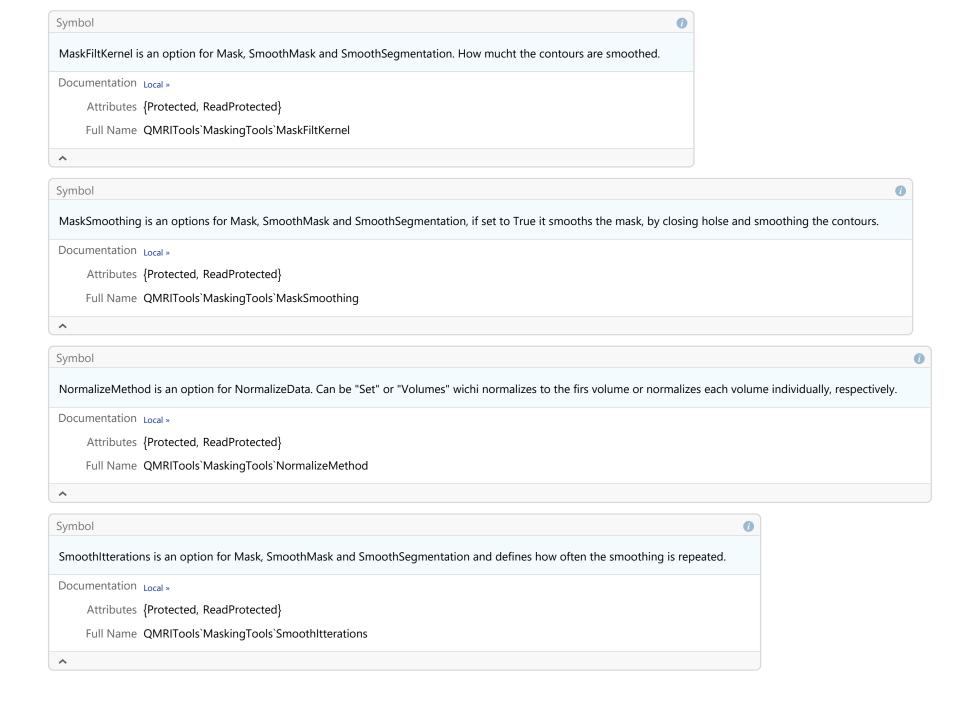
Full Name QMRITools`MaskingTools`SmoothSegmentation

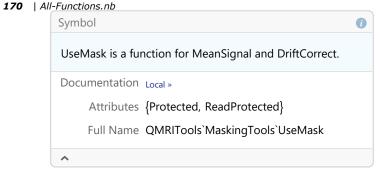


### **Options**





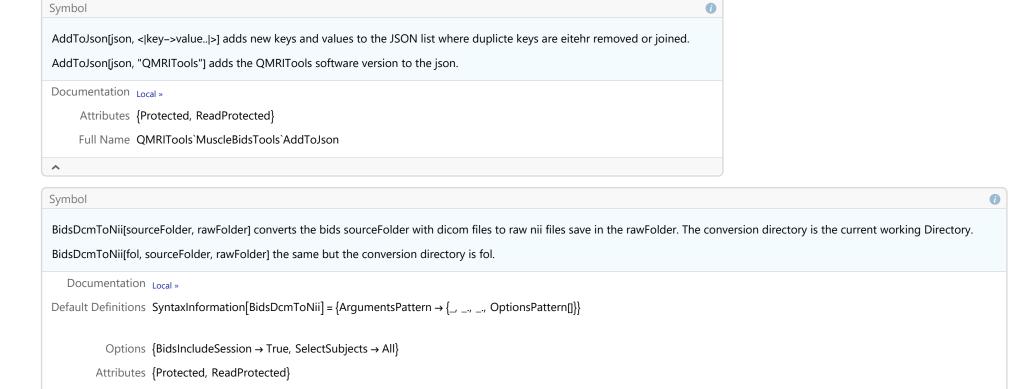


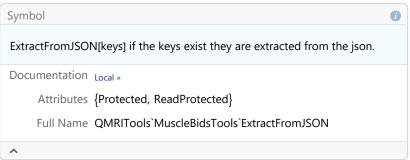


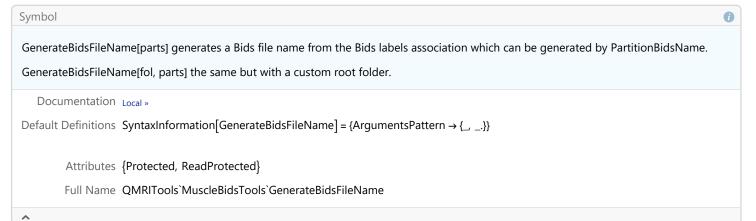
Full Name QMRITools`MuscleBidsTools`BidsDcmToNii

## MuscleBidsTools

#### **Functions**

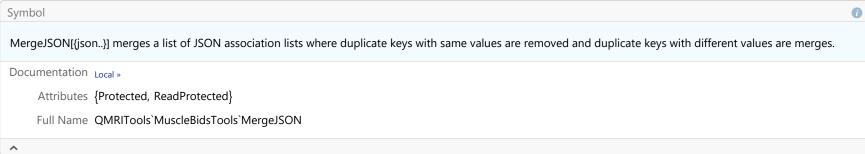








Symbol



0

```
MuscleBidsConvert[niiFol, discription] converts all nii data in the niiFol subfolder "raw" to Muscle-Bids based on the data discription.

Example discription:

{"Label" -> "DIXON", "Type" -> "megre", "Class" -> "Stacks", "Overlap" -> 5}

{"Label" -> "DIXON", "Type" -> "megre", "Class" -> "Stacks"}

{"Label" -> "DTI", "Type" -> "dwi", "Class" -> "Stacks", "Overlap" -> 5, "Suffix" -> "dti"}

{"Label" -> "DIXON", "Type" -> "megre"}.

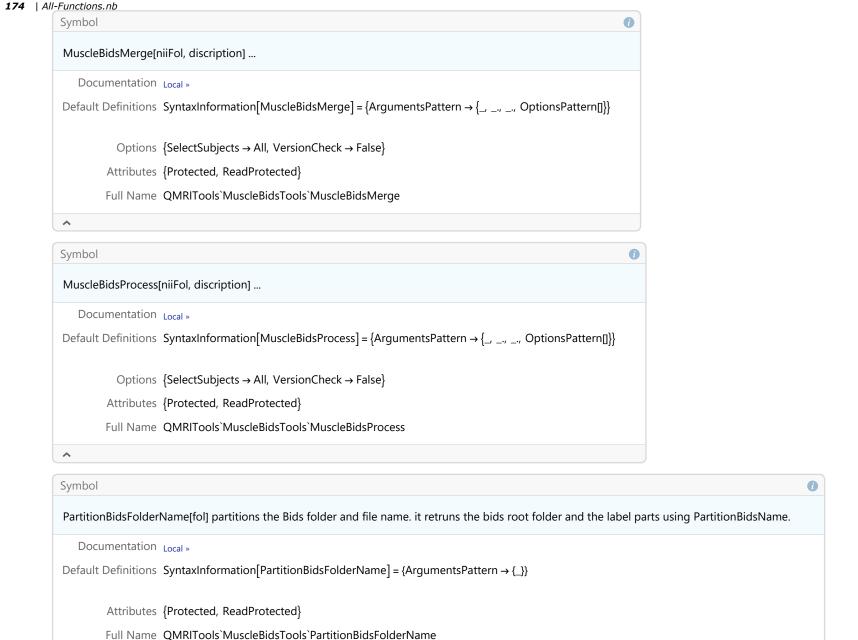
Documentation Local »

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

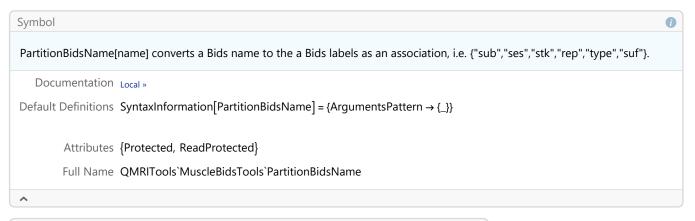
Options {DeleteAfterConversion → True, SelectSubjects → All}

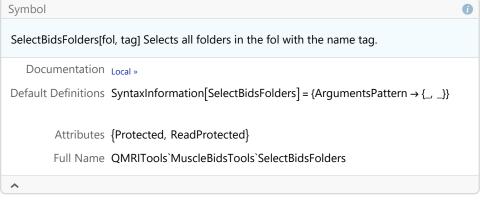
Attributes {Protected, ReadProtected}

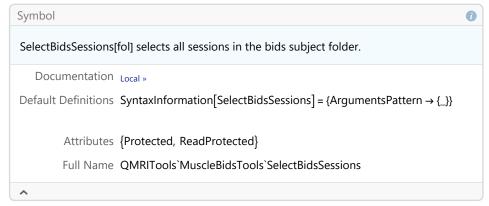
Full Name QMRITools'MuscleBidsTools'MuscleBidsConvert
```

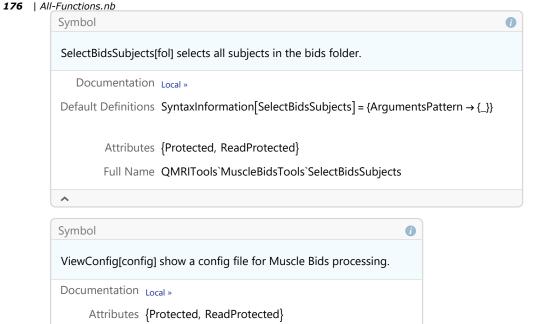


 $\wedge$ 









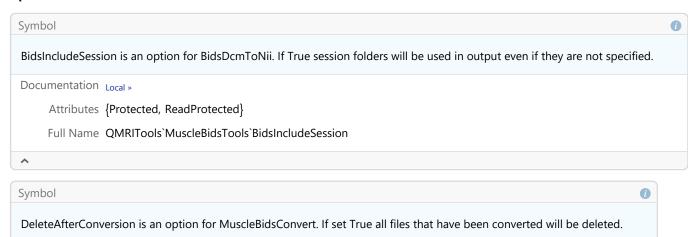
Full Name QMRITools`MuscleBidsTools`ViewConfig

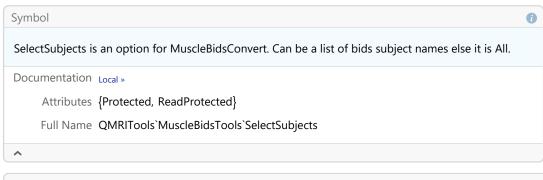
### **Options**

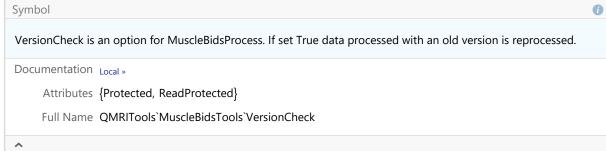
Documentation Local »

Attributes {Protected, ReadProtected}

Full Name QMRITools`MuscleBidsTools`DeleteAfterConversion

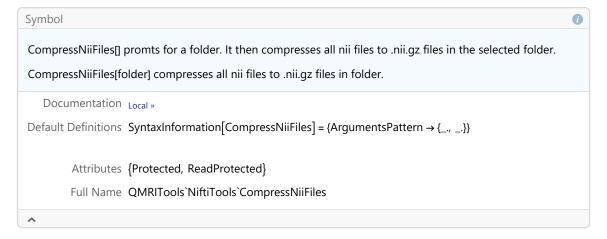


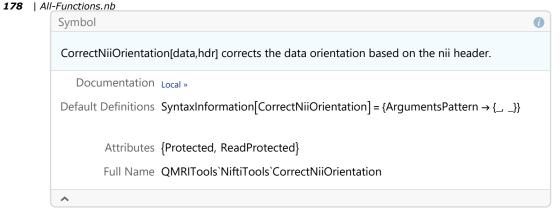


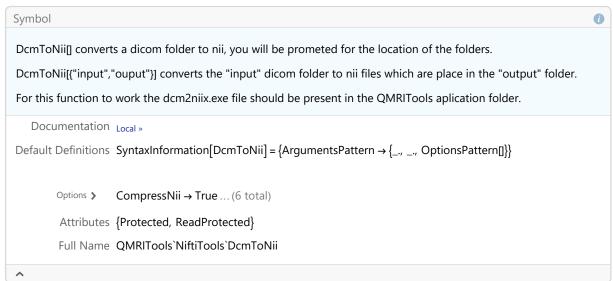


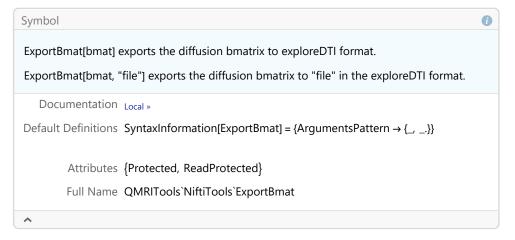
## **NiftiTools**

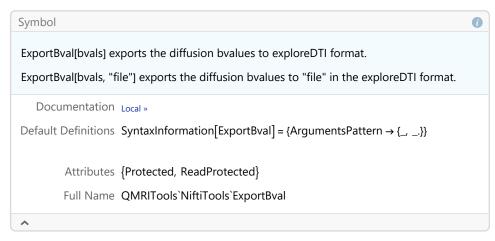
#### **Functions**

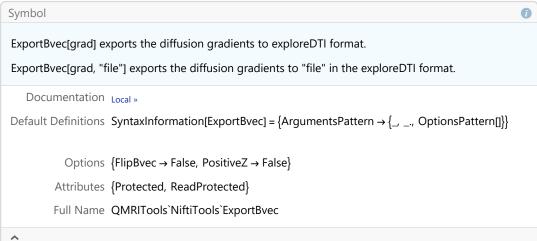


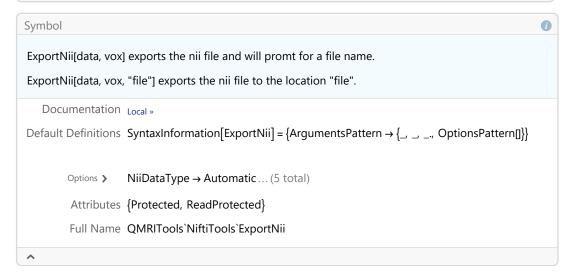


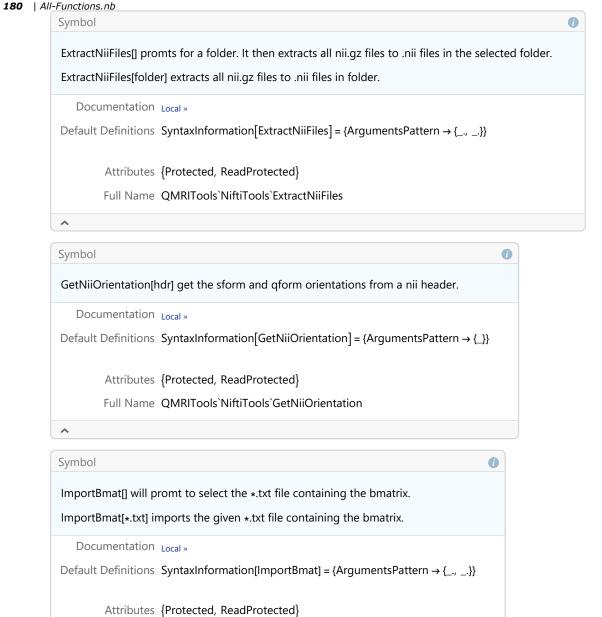




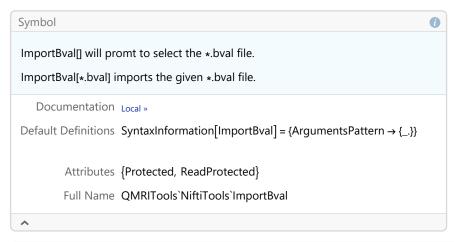


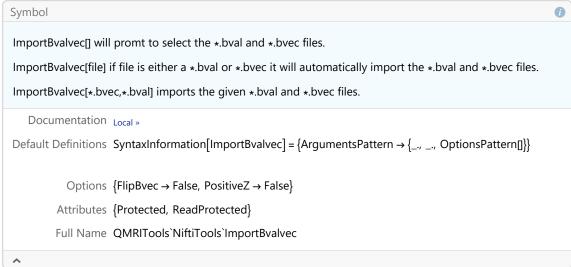


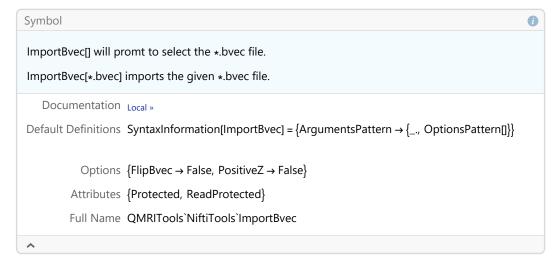




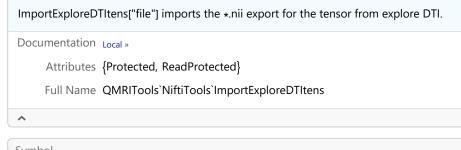
Full Name QMRITools`NiftiTools`ImportBmat

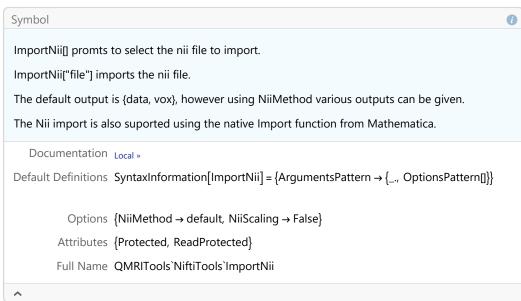


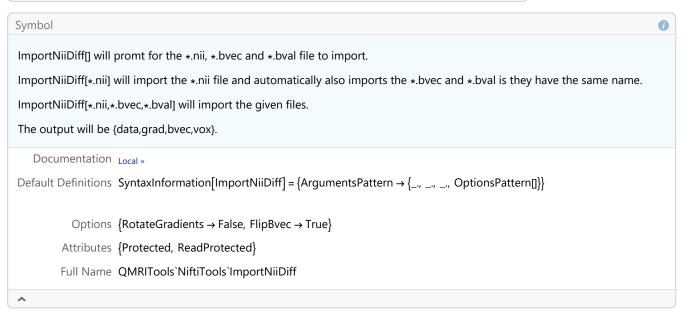


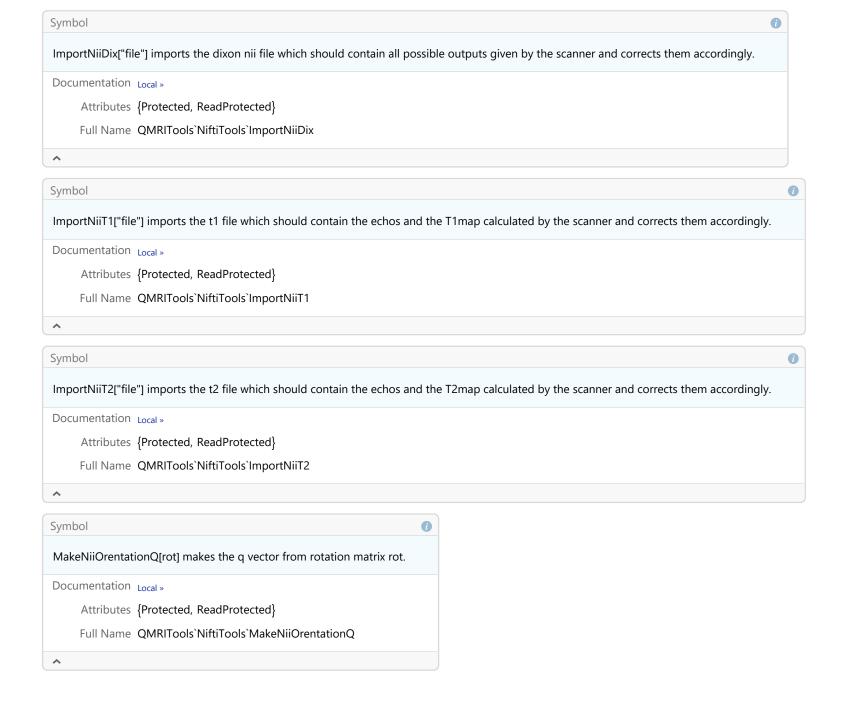


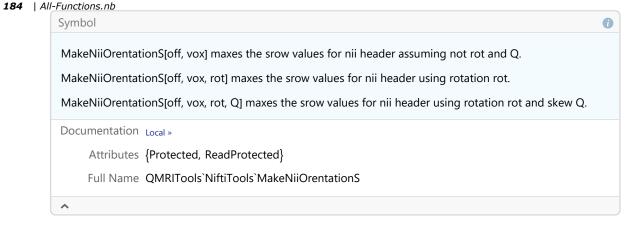


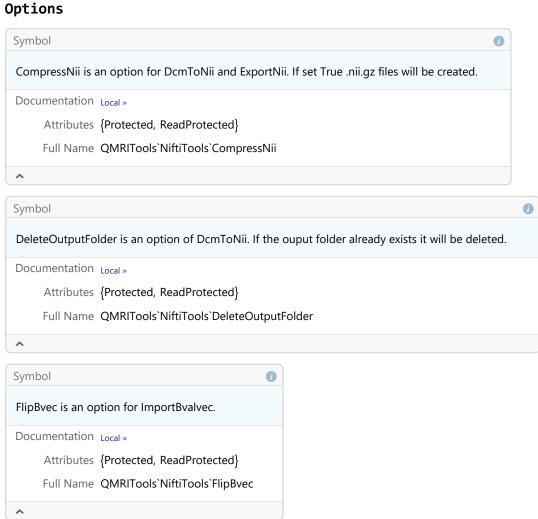


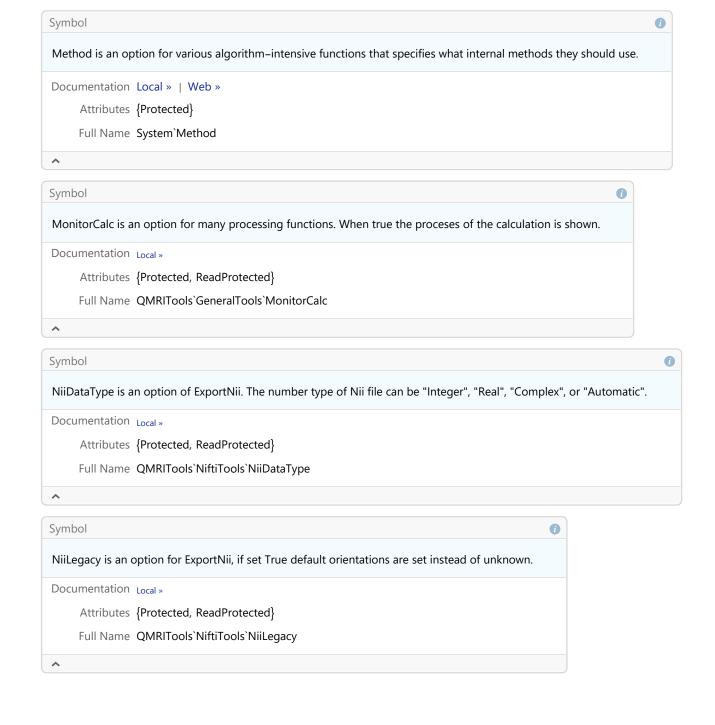


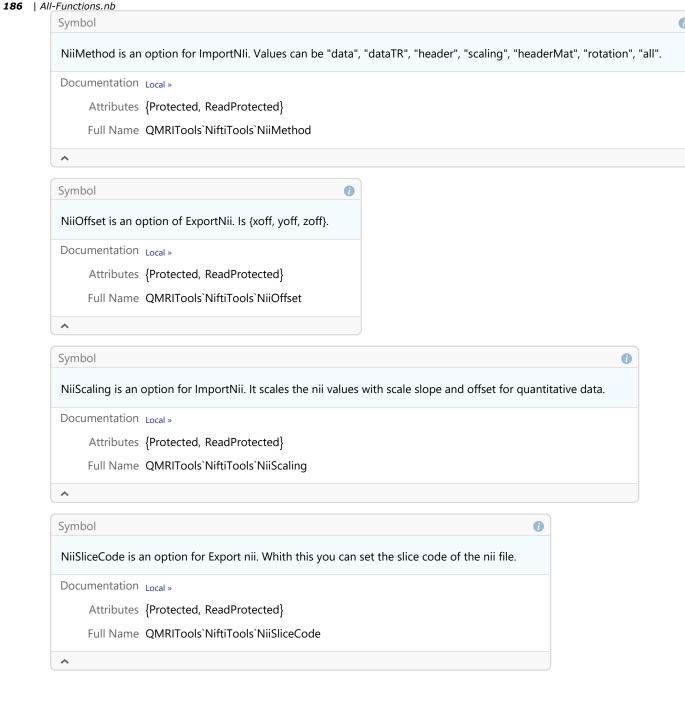


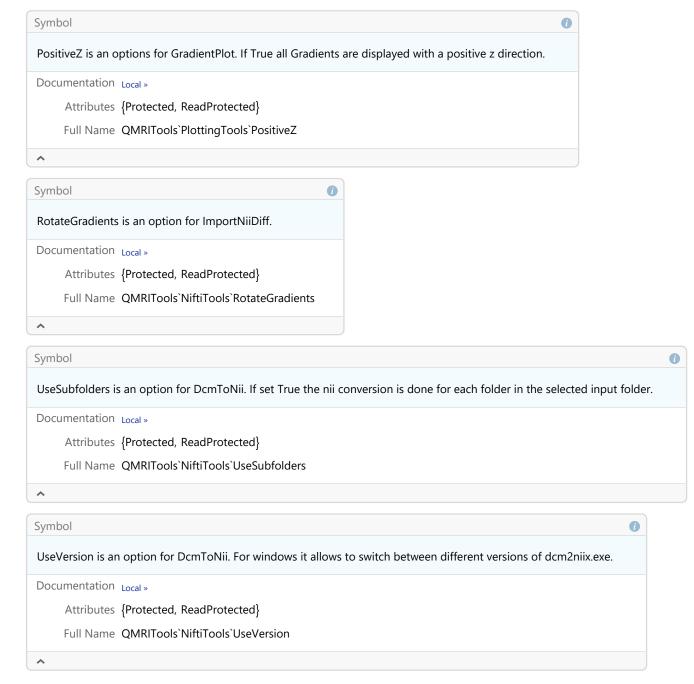






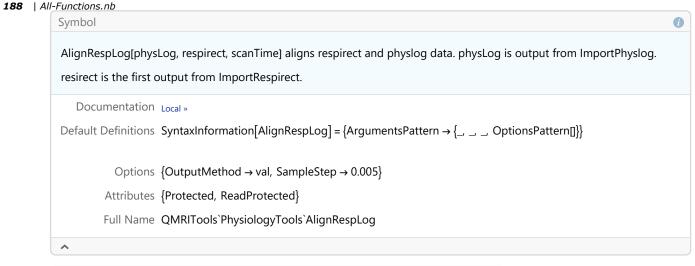


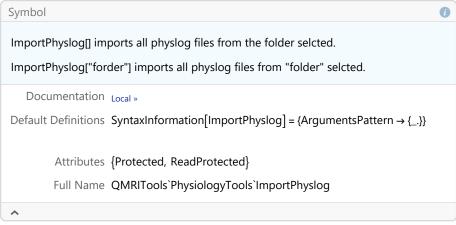


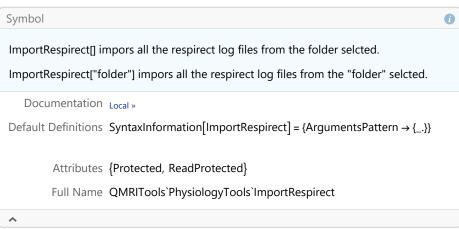


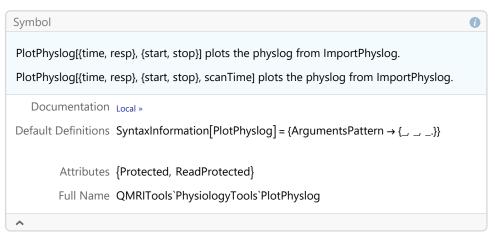
# PhysiologyTools

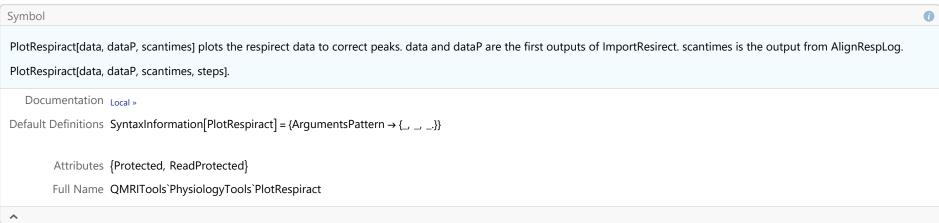
**Functions** 



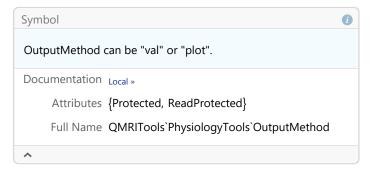


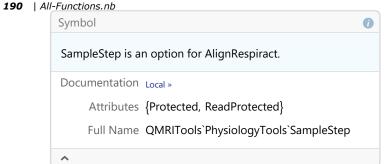






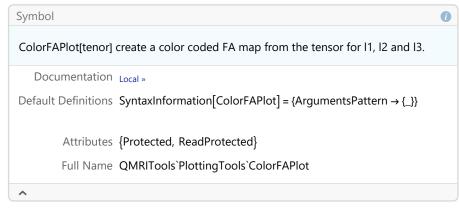
#### **Options**

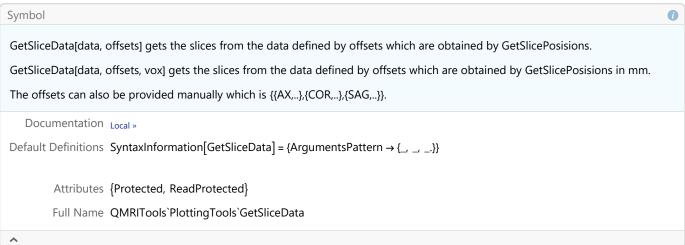




## PlottingTools

#### **Functions**





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 → False, DropSlices → {1, 1, 1}, PeakNumber → {1, 1, 2}}

Attributes {Protected, ReadProtected}

Full Name QMRITools`PlottingTools`GetSlicePositions

GradientPlot[bvec, bval] plots the given bvec with position of the gradients scaled according to the bval.

Documentation Local >>

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

Options > PlotSpace → bspace ... (4 total)

Attributes {Protected, ReadProtected}

Full Name QMRITools`PlottingTools`GradientPlot

ListSpherePlot[points] plots 3D points as spheres.

Documentation Local »

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

Options {SphereSize → 2, SphereColor → Automatic}

Attributes {Protected, ReadProtected}

Full Name QMRITools`PlottingTools`ListSpherePlot

Symbol

 $\wedge$ 

MakeSliceImages[imgData] generates images from the imgData which is obtained form GetSliceData.

MakeSliceImages[imgData, vox] generates images from the imgData which is obtained form GetSliceData, vox is used for the correct aspect ratio of the images.

MakeSliceImages[imgData, {labData, labels}] generates images from the imgData which is obtained

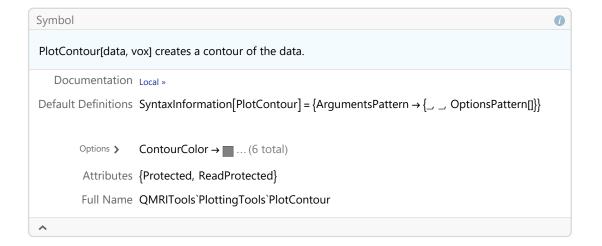
form GetSliceData with an overlay of the segmentations in labData, which can also be obtained using GetSliceData on the segmentations.

labels should be the label numbers used in the original segmentation (to allow correct scaling between slices).

MakeSliceImages[imgData, {labData, labels},vox] generates images from the imgData which is obtained form GetSliceData with an overlay

of the segmentations in labData, which can also be obtained using GetSliceData on the segmentations, vox is used for the correct aspect ratio of the images.

Documentation Local » Default Definitions SyntaxInformation[MakeSliceImages] = {ArgumentsPattern → {\_\_\_,\_\_, OptionsPattern[]}} PlotRange → Automatic ... (4 total) Attributes {Protected, ReadProtected} Full Name QMRITools`PlottingTools`MakeSliceImages



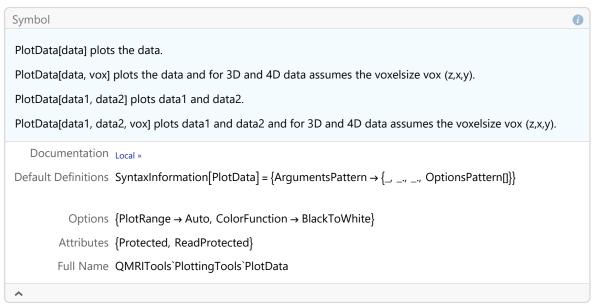
```
PlotCorrection[w] plots deformation vectors w {w1,w2...} generated by Registration2D and Registration3D for multiple datasets or registration steps.

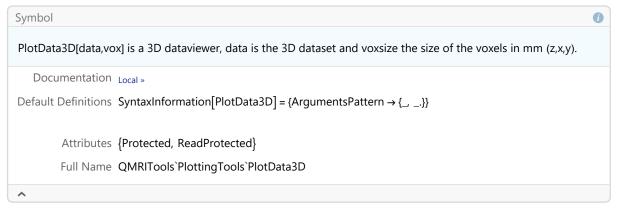
Documentation Local »

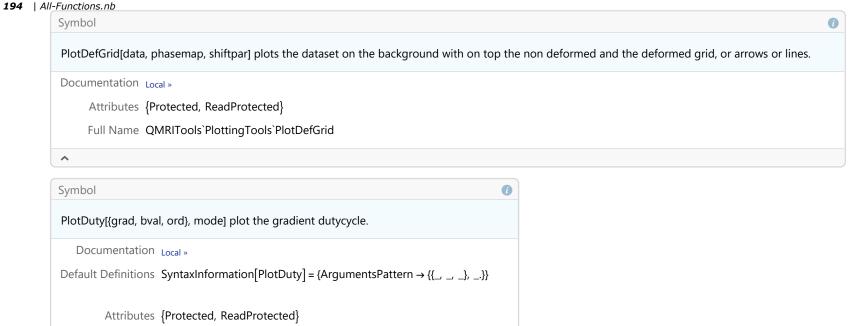
Default Definitions SyntaxInformation[PlotCorrection] = {ArgumentsPattern → {_}}

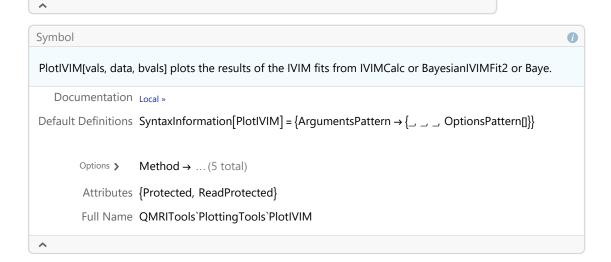
Attributes {Protected, ReadProtected}

Full Name QMRITools`PlottingTools`PlotCorrection
```

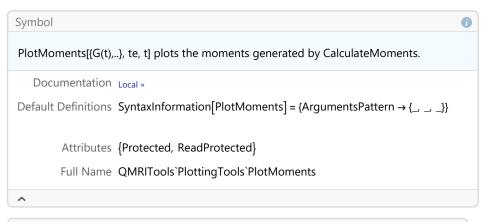


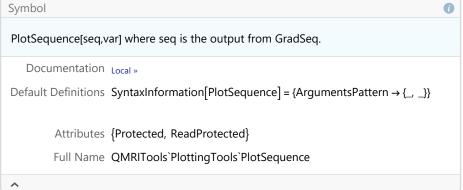




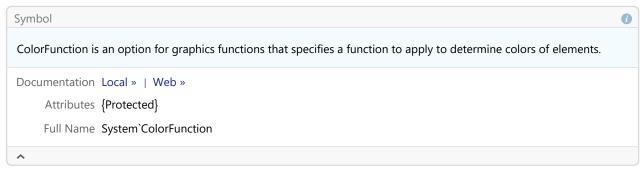


Full Name QMRITools`PlottingTools`PlotDuty





### **Options**

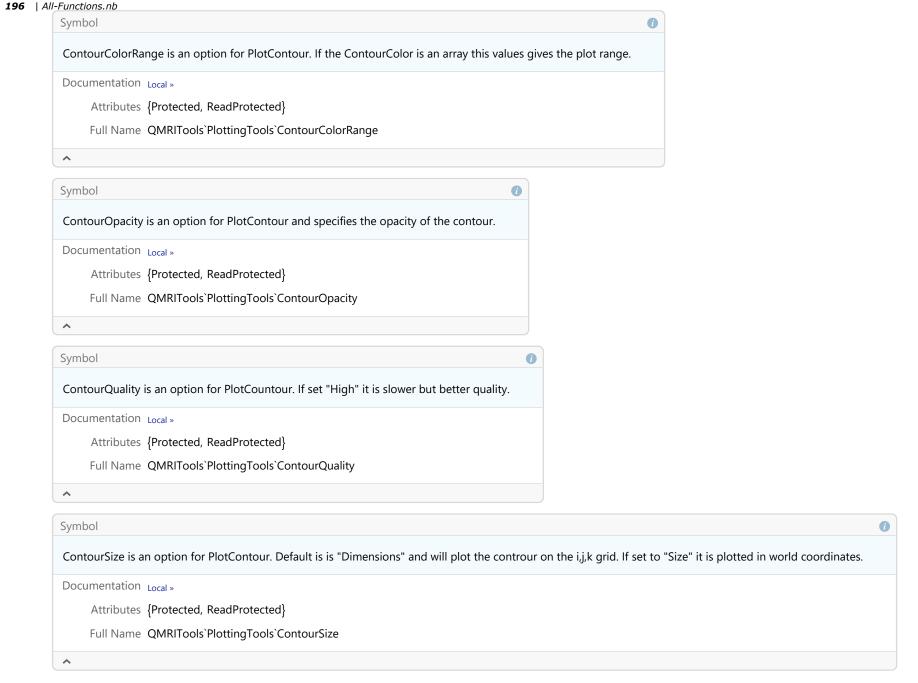


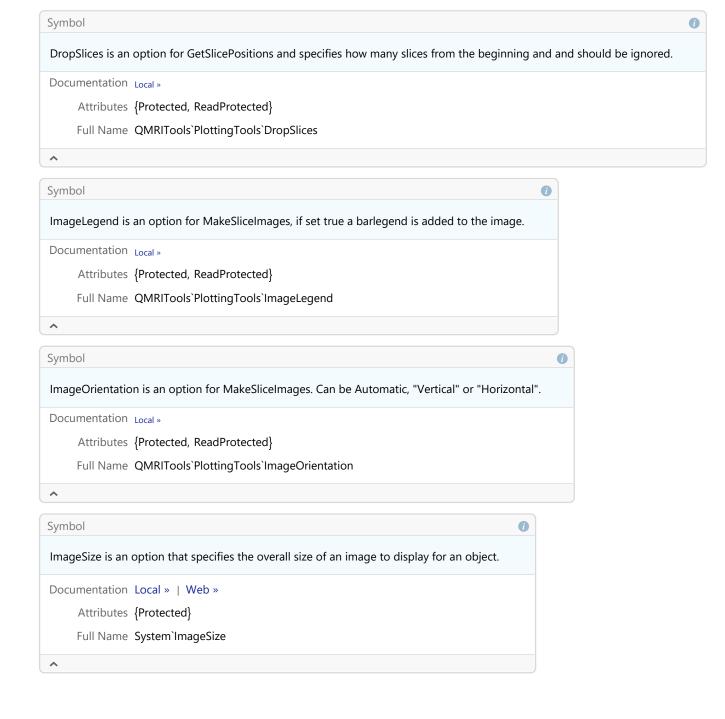
```
ContourColor is an option for PlotContour. It specifies the contour color and can be a single color or an array with the same dimensions as the data.

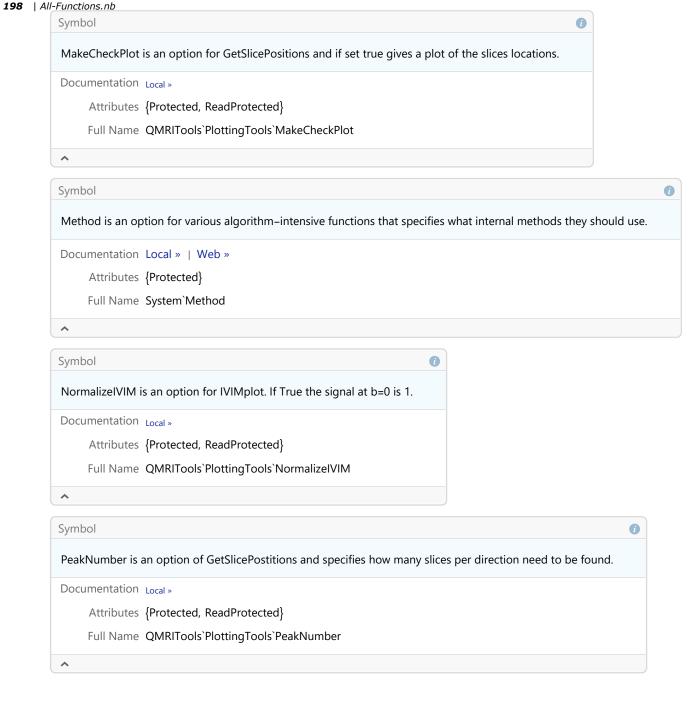
Documentation Local »

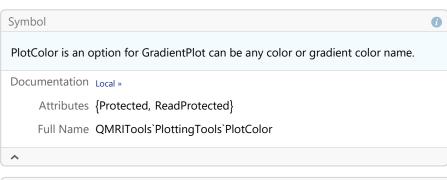
Attributes {Protected, ReadProtected}

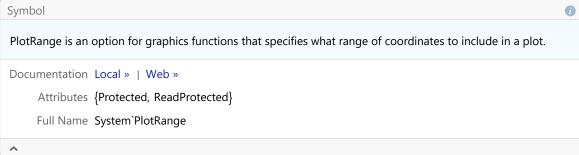
Full Name QMRITools`PlottingTools`ContourColor
```

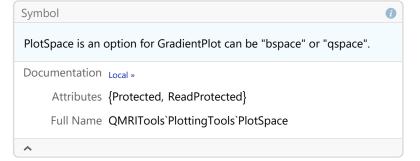


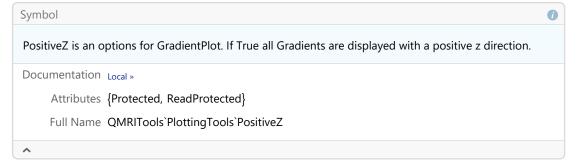


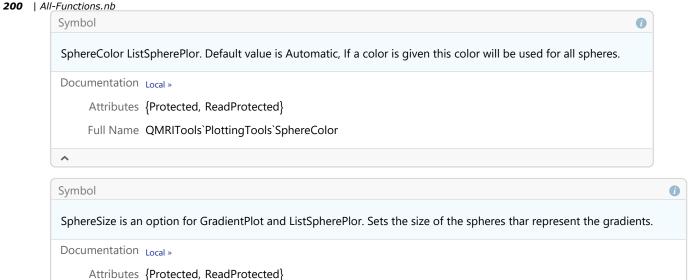










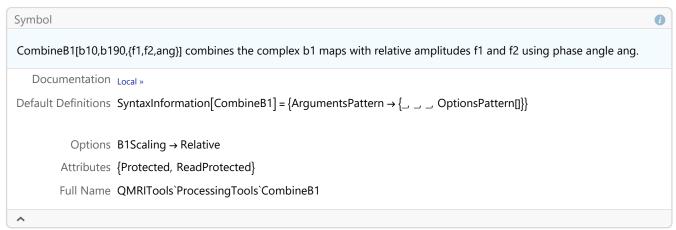


### ProcessingTools

Full Name QMRITools`PlottingTools`SphereSize

#### **Functions**





```
CorrectJoinSetMotion[[{dat1, dat2, ...}, vox, over] motion correts multiple sets with overlap. Over is the number of slices overlap between stes. A Translation registration is performed.

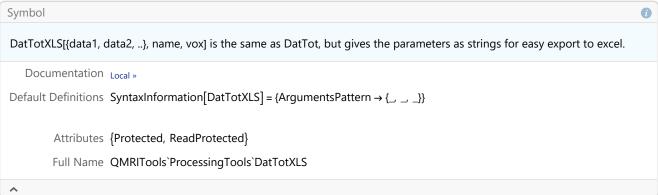
Documentation Local »

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

Options {JoinSetSplit → True, PaddOverlap → 2, MonitorCalc → True}

Attributes {Protected, ReadProtected}

Full Name QMRITools'ProcessingTools'CorrectJoinSetMotion
```



ErrorPlot[data, xdata] plots a errorplot of the data where the first dim of the data is the xrange which matches the xdata list.

ErrorPlot[data, xdata, range] similar with a given y range.

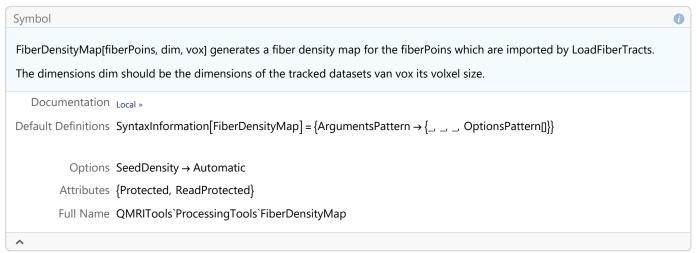
Documentation Local >>

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

Options > ColorValue → {\_\_, \_\_\_} ... (5 total)

Attributes {Protected, ReadProtected}

Full Name QMRITools`ProcessingTools`ErrorPlot



FiberLengths[fpoints, flines] calculates the fiber lenght using the output from LoadFiberTacts.

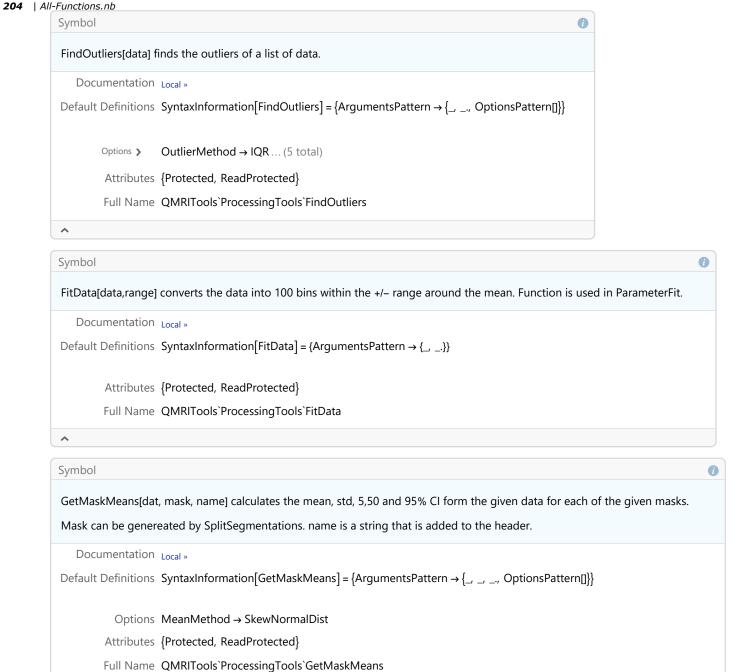
FiberLengths[fpoints, flines]] calculates the fiber lenght using the output from LoadFiberTacts.

Documentation Local »

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

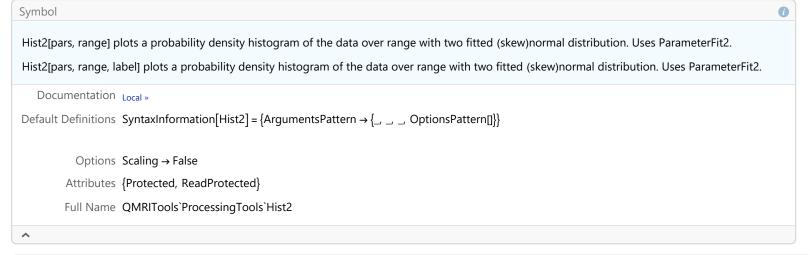
Attributes {Protected, ReadProtected}

Full Name QMRITools`ProcessingTools`FiberLengths

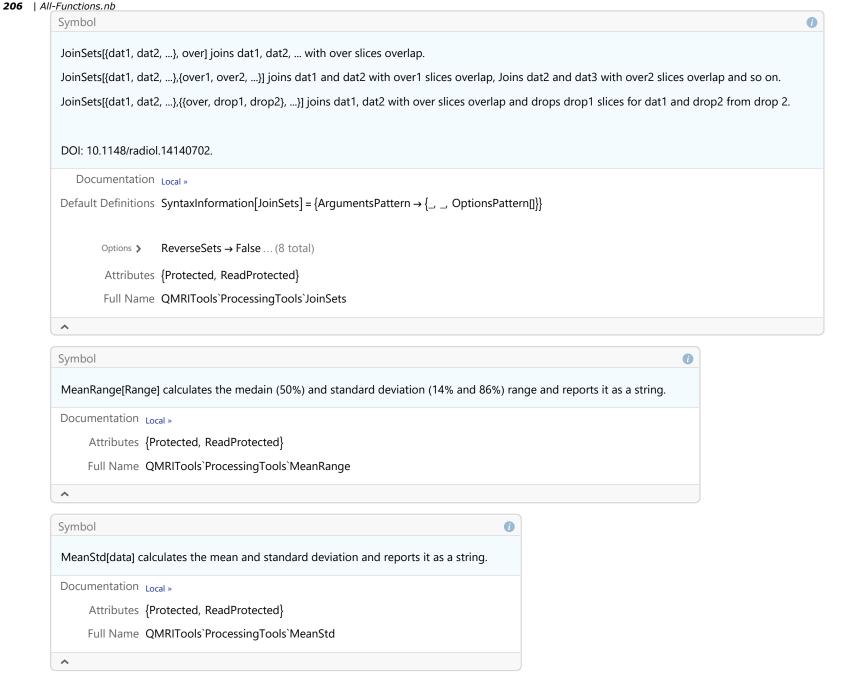


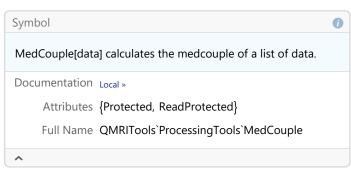
^

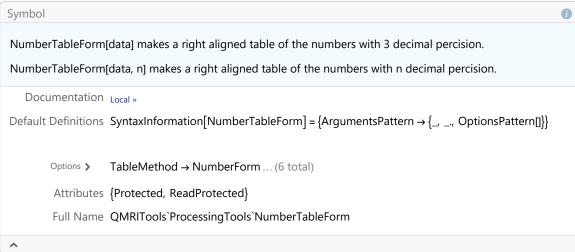
```
Symbol
                                                                                                                                                                                                                     0
Hist[data, range] plots a probability density histogram of the data from xmin 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] = {ArgumentsPattern \rightarrow {_, _, _, OptionsPattern[]}}
                     ColorValue \rightarrow \{\{\blacksquare, \square\}, \blacksquare, \blacksquare, \dots, [5 \text{ total}\}\}
         Options >
         Attributes {Protected, ReadProtected}
         Full Name QMRITools`ProcessingTools`Hist
^
```

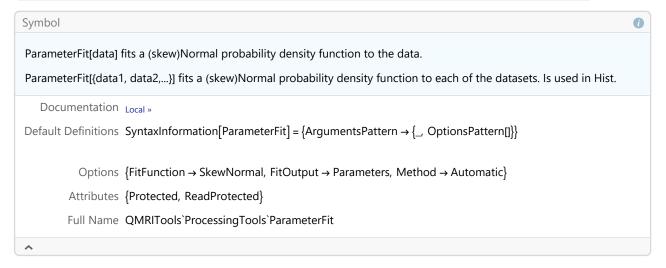


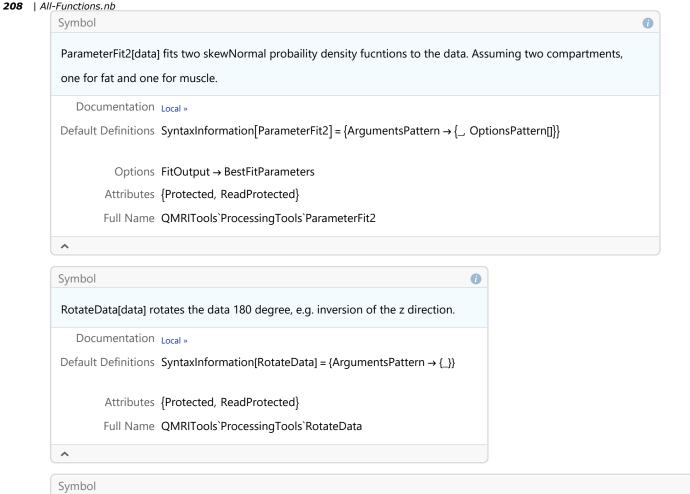
```
Symbol
InvertDataset[data] inverts the data along the x y and z axes. In other words it is rotated aroud the origin such that (x,y,z)=(-x,-y,-z) and (0,0,0)=(0,0,0).
Documentation Local »
     Attributes {Protected, ReadProtected}
     Full Name QMRITools`ProcessingTools`InvertDataset
```

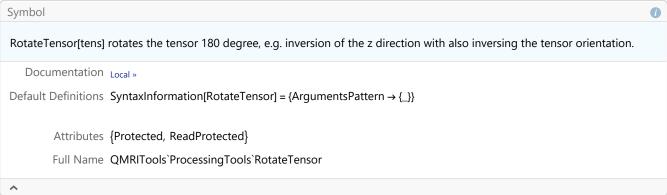




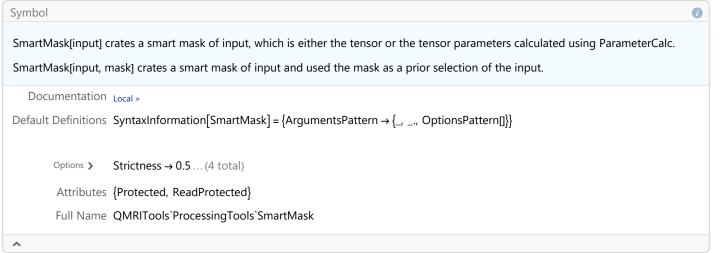


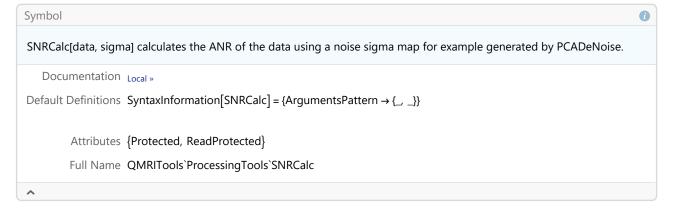


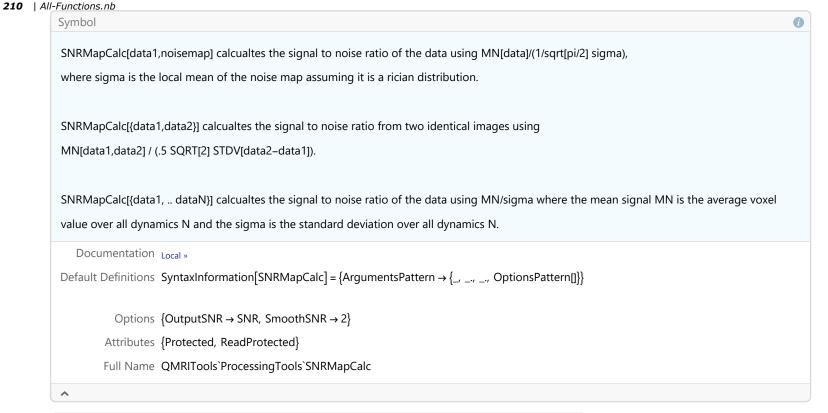


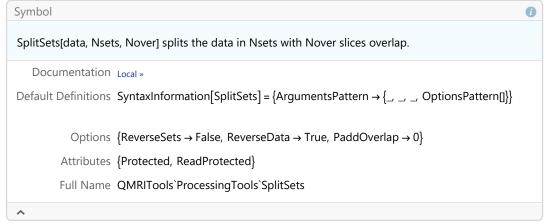




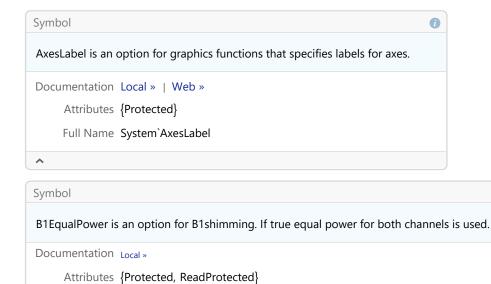




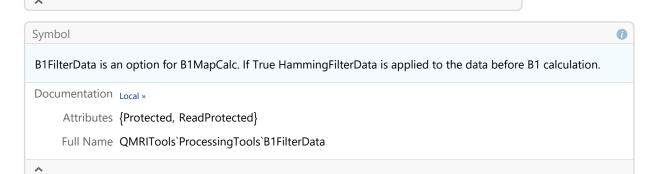


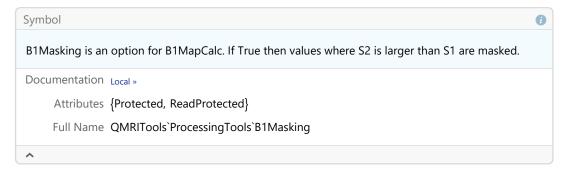


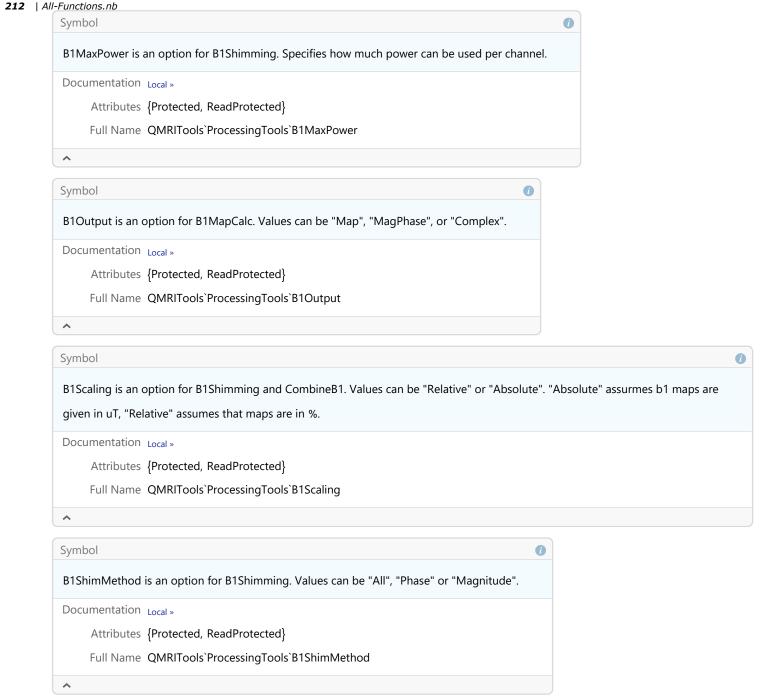
#### **Options**

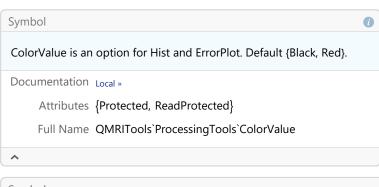


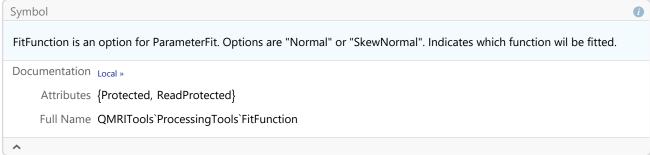
Full Name QMRITools`ProcessingTools`B1EqualPower

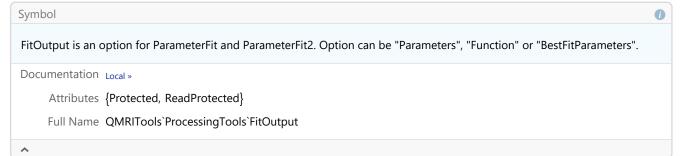




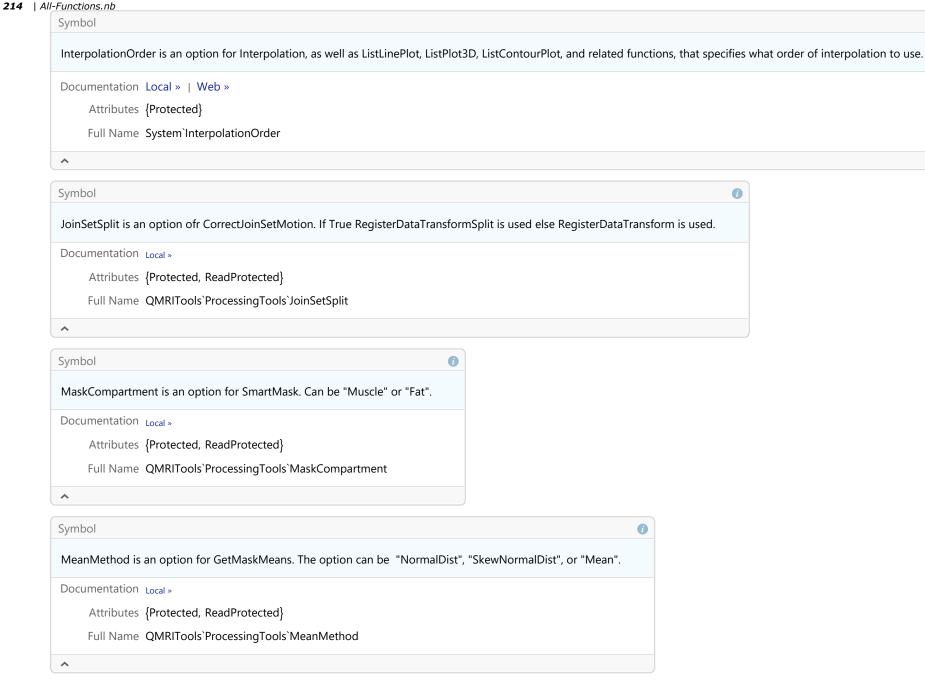


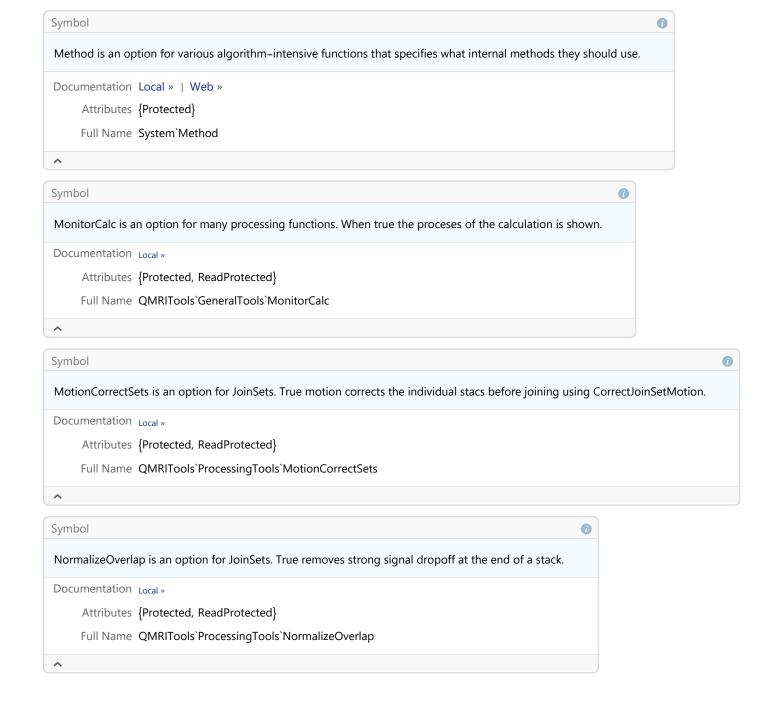


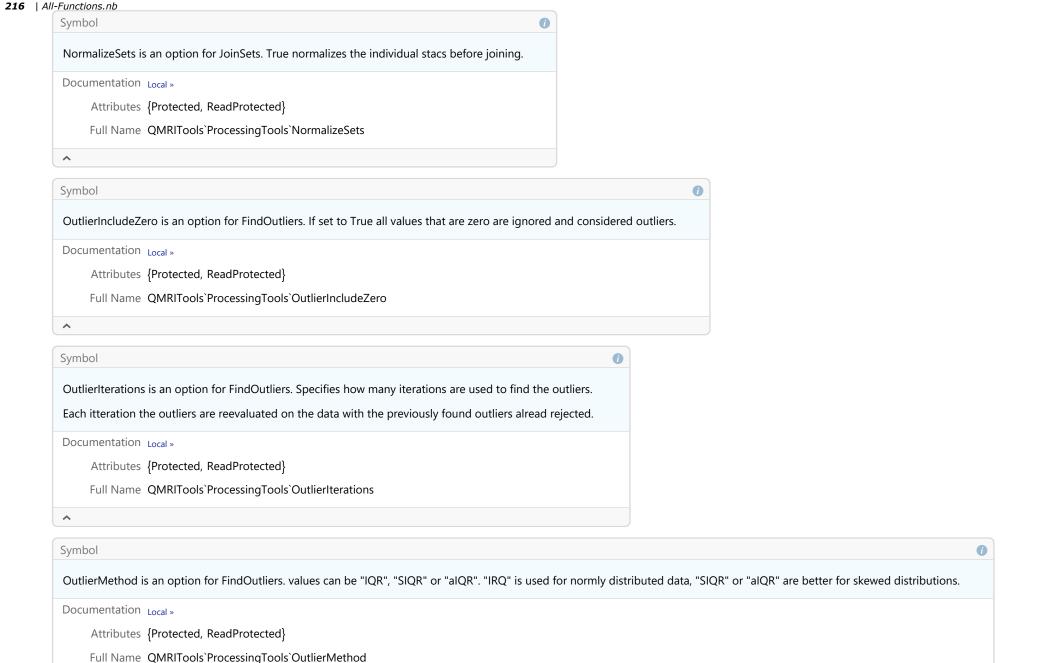




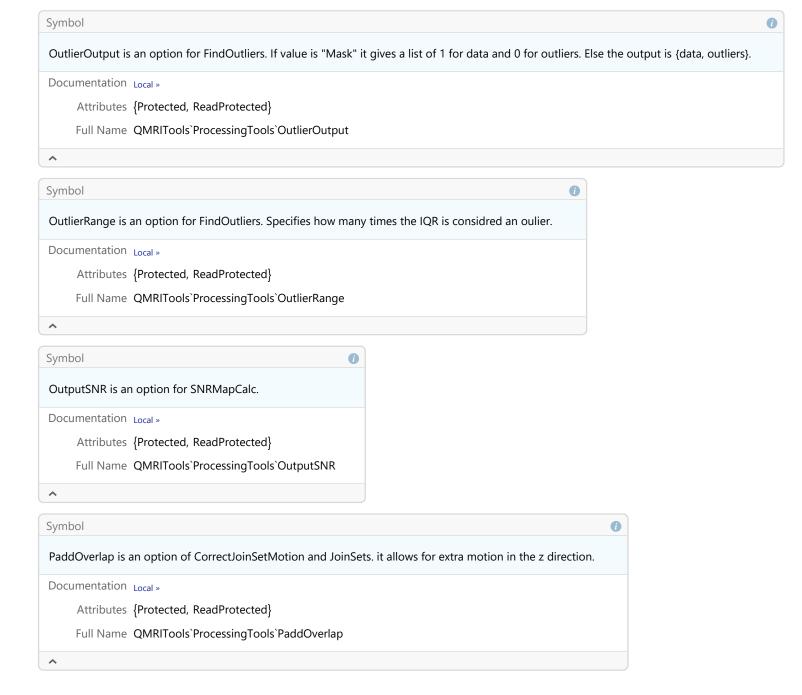


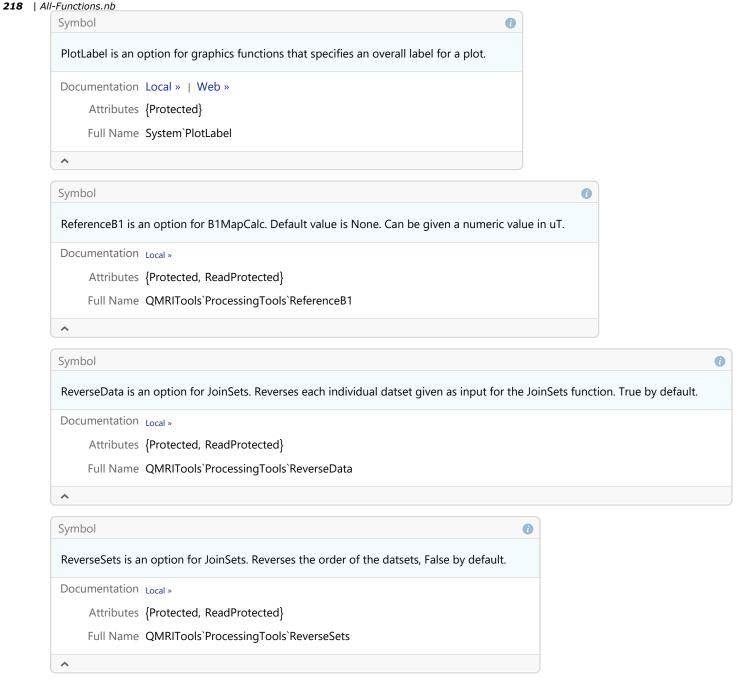


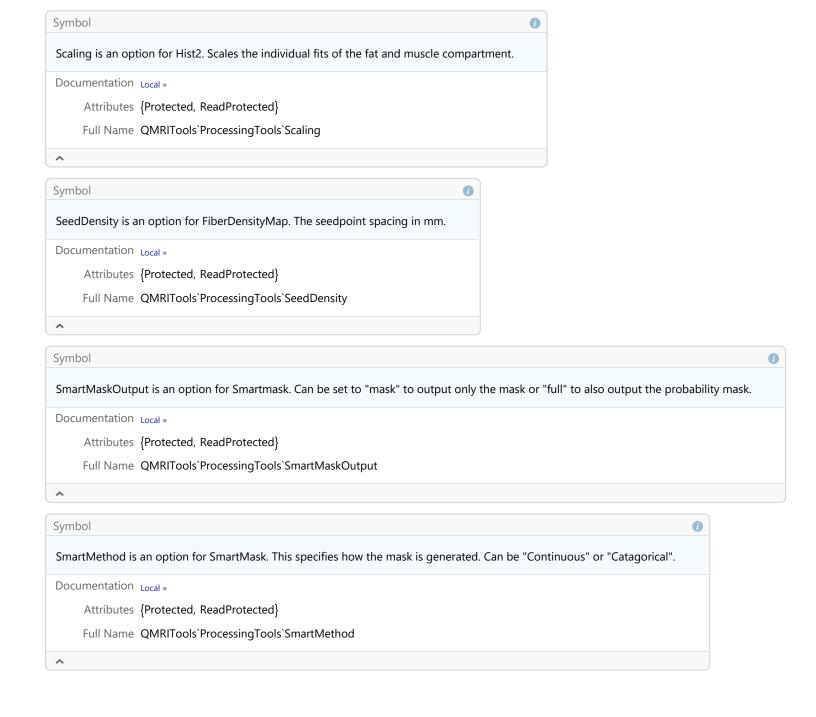


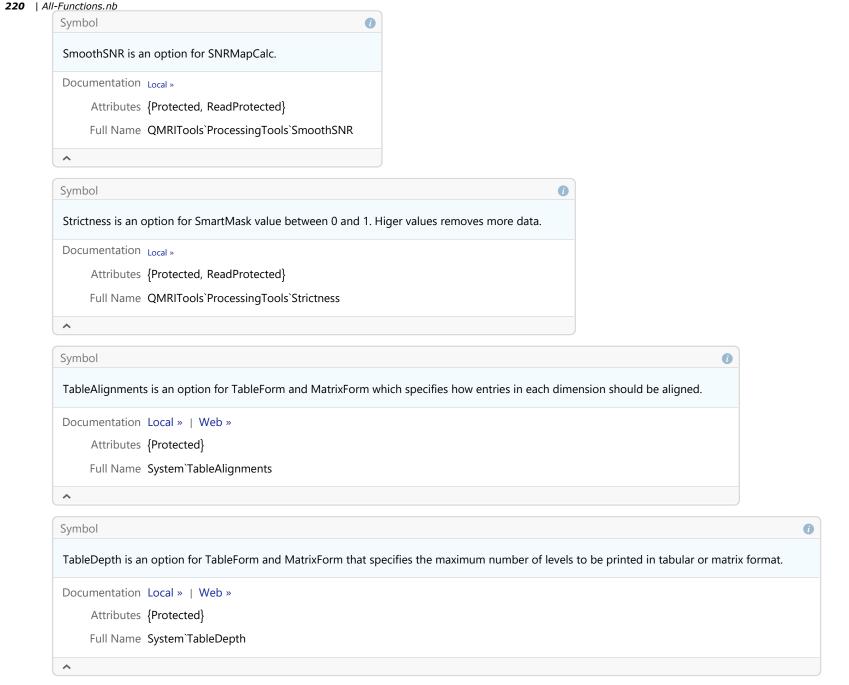


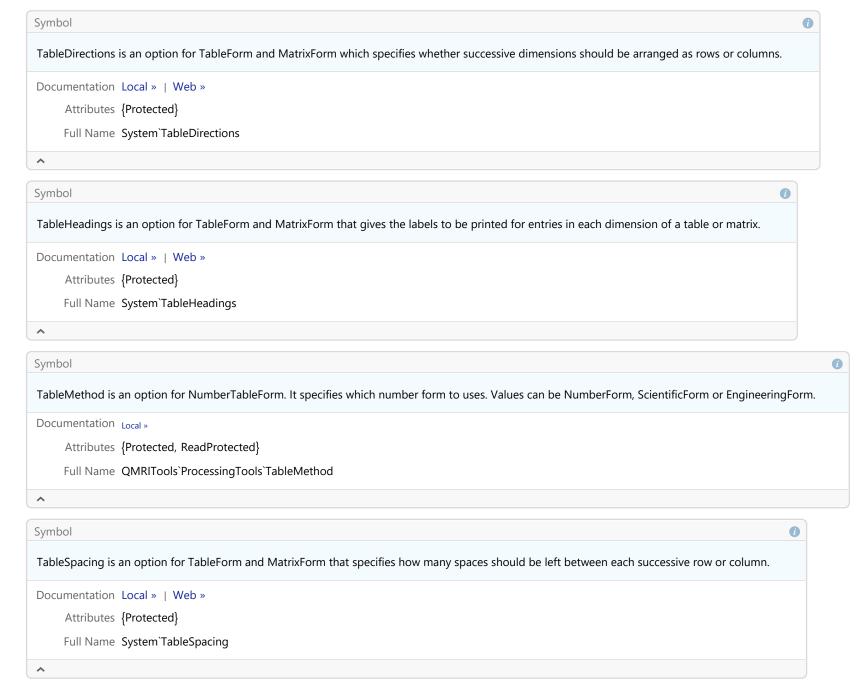
 $\wedge$ 











# ReconstructionTools

## **Functions**

Symbol

CoilCombine[sig] combines the coil signals sig. Where sig is {nCoils, ...}.

CoilCombine[sig, cov] combines the coil signals sig. Where sig is {nCoils, ...} and cov the complex noise correlation matrix.

CoilCombine[siq, cov, sens] combines the coil signals sig. Where sig is {nCoils, ...} and cov the complex noise correlation matrix and sense the coils sensitivitys.

Possible coil combination methods are "Sum", "RootSumSqaures", "RoemerEqualNoise", "RoemerEqualSignal", "WSVD".

RootSumSquares needs the signal. Can be performed with and without the noise covaricance matrix

RoemerEqualNoise needs the signal and the noise covaricance matrix. Can be performed with

and without the sense data, without sense data the sensisity is estimated using the singal and the RSS reconstruction of the signa.

RoemerEqualSignal needs the signal and the noise covaricance matrix and the sense data.

WSVD needs the signal and the noise covariance matrix.

Documentation Local »

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

Options {Method → RoemerEqualNoise, SenseRescale → False}

Attributes {Protected, ReadProtected}

Full Name QMRITools'ReconstructionTools'CoilCombine

^

### Symbol



CoilWeightedRecon[kspace, noise, head] performs reconstuction of raw MS2D MRI data. The input kspace, noise and head are obtained using ReadListData.

The coil combination Methods can be "Roemer" or "RSS".

Documentation Local »

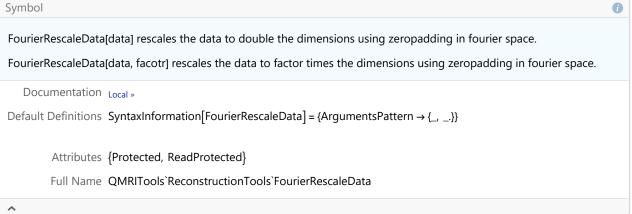
Default Definitions SyntaxInformation[CoilWeightedRecon] = {ArgumentsPattern → {\_ \_ \_ \_ \_ \_ \_ , OptionsPattern[]}}

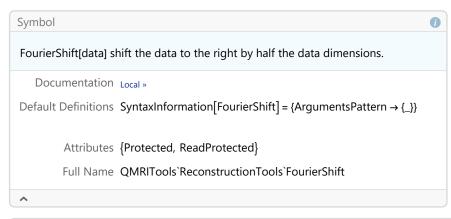
Options > EchoShiftData  $\rightarrow 0 \dots (6 \text{ total})$ 

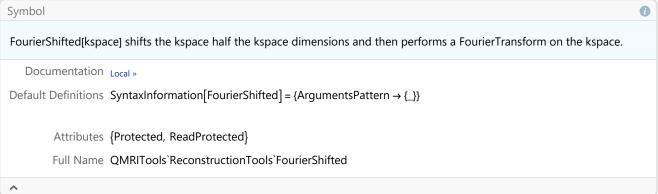
Attributes {Protected, ReadProtected}

Full Name QMRITools`ReconstructionTools`CoilWeightedRecon

```
Symbol
CoilWeightedReconCSI[kspace, noise, head] performs reconstuction of raw 3DCSI data. The input kspace, noise and head are obtained using ReadListData.
The coil combination Methods can be "Roemer" or "WSVD".
   Documentation Local »
Default Definitions SyntaxInformation[CoilWeightedReconCSI] = {ArgumentsPattern → {_, _, _, _, OptionsPattern[]}}
                   HammingFilter → False ... (5 total)
        Options >
        Attributes {Protected, ReadProtected}
        Full Name QMRITools`ReconstructionTools`CoilWeightedReconCSI
Symbol
                                                                                                                                              0
DeconvolveCSIdata[spectra] deconvolves the CSI spectra after HammingFilterCSI to revert the blurring of the hammingfiltering.
DeconvolveCSIdata[spectra, ham] deconvolves the CSI spectra with the acquired weighting ham to revert the blurring of the kspace weighting.
   Documentation Local »
Default Definitions SyntaxInformation[DeconvolveCSIdata] = {ArgumentsPattern → {_, _, OptionsPattern[]}}
          Options {WienerRegularization → 0.007, DeconvolutionMethod → Wiener}
        Attributes {Protected, ReadProtected}
        Full Name QMRITools'ReconstructionTools'DeconvolveCSIdata
                                                                                                                                        0
Symbol
FourierKspace2D[kspace,head] performs a 2D reconstruction of 2D kspace data. Where kspace and head are generated by ReadListData.
   Documentation Local »
Default Definitions SyntaxInformation[FourierKspace2D] = \{ArgumentsPattern \rightarrow \{\_, \_, \_\}\}
        Attributes {Protected, ReadProtected}
        Full Name QMRITools`ReconstructionTools`FourierKspace2D
\wedge
```







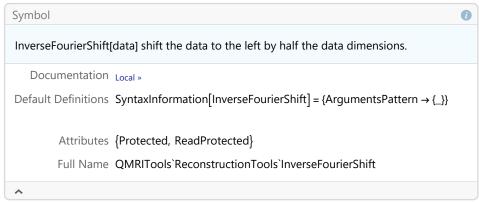
HammingFilterCSI[kspace] apllies a Hammingfilter to the k-space data. The data can be can be 1D, 2D or 3D, the spectral dimensions is the last dimensions (x,y,z, spectra).

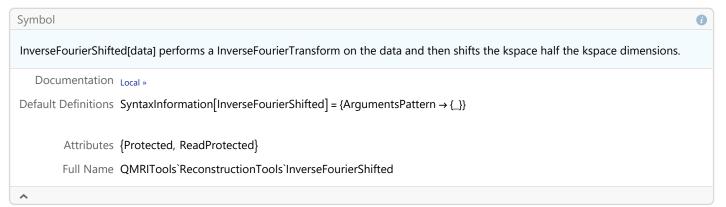
Documentation Local >>

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

Attributes {Protected, ReadProtected}

Full Name QMRITools`ReconstructionTools`HammingFilterCSI





```
MakeHammingFilter[xdim] makes a 1D HammingKernel for filtering k-space.

MakeHammingFilter[xdim] makes a 1D HammingKernel for filtering k-space.

MakeHammingFilter[xdim, ydim] makes a 2D HammingKernel for filtering k-space in 2D CSI data of size {xdim, ydim}.

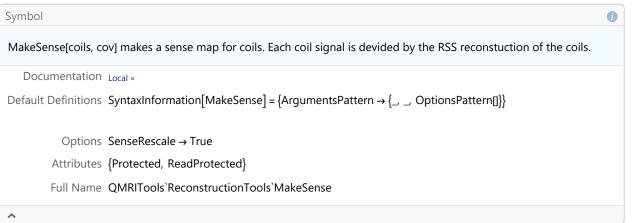
MakeHammingFilter[xdim, ydim, zdim] makes a 3D HammingKernel for filtering k-space in 3D CSI data of size {xdim, ydim, zdim}.

Documentation Local *

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

Attributes {Protected, ReadProtected}

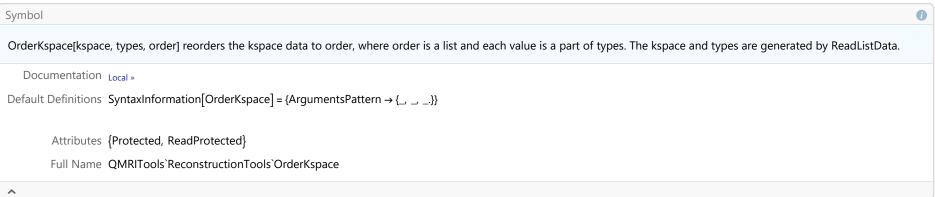
Full Name QMRITools`ReconstructionTools`MakeHammingFilter
```

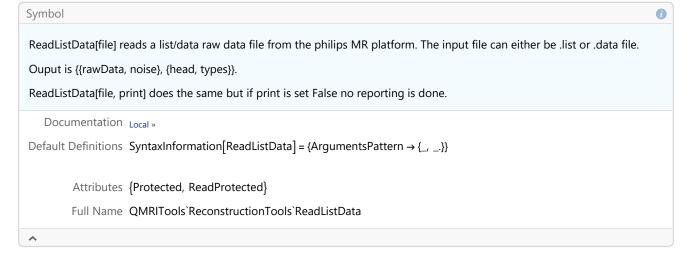


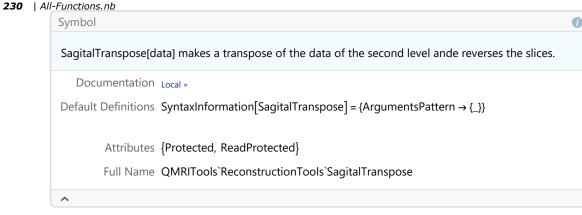
 $\wedge$ 

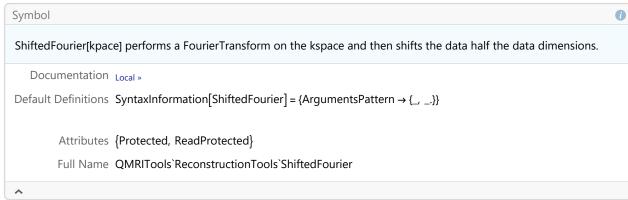
228 | All-Functions.nb Symbol MeanType[kspace, types, type] calcualtes the Mean of the kspace data on type, where type is part of types. The kspace and types are generated by ReadListData. MeanType[{kspace, types}, type] calcualtes the Mean of the kspace data on type, where type is part of types. MeanType[kspace, types, {type,...}] calcualtes the Mean of the kspace data on each of the list type, where type is part of types. Output is {kspace, types}. Documentation Local » Default Definitions SyntaxInformation[MeanType] = {ArgumentsPattern → {\_, \_, \_, \_.}} Attributes {Protected, ReadProtected} Full Name QMRITools`ReconstructionTools`MeanType ^ Symbol 0 NoiseCorrelation[noise] calculates the noise correlation matrix, noise is {nrCoils, noise Samples}. Documentation Local » Default Definitions SyntaxInformation[NoiseCorrelation] =  $\{ArgumentsPattern \rightarrow \{\_\}\}$ Attributes {Protected, ReadProtected} Full Name QMRITools'ReconstructionTools'NoiseCorrelation ^ Symbol NoiseCovariance[noise] calculates the noise covariance matrix, noise is {nrCoils, noise Samples}. Documentation Local » Default Definitions SyntaxInformation[NoiseCovariance] =  $\{ArgumentsPattern \rightarrow \{\_\}\}$ Attributes {Protected, ReadProtected}

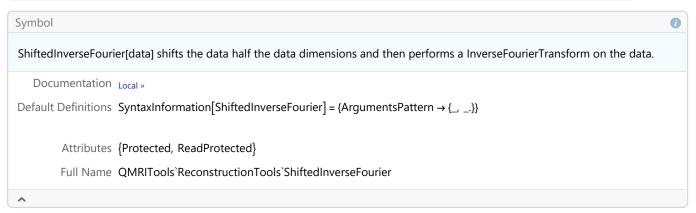
Full Name QMRITools'ReconstructionTools'NoiseCovariance











```
TotalType[kspace, types, type] calcualtes the Total of the kspace data on type, where type is part of types. The kspace and types are generated by ReadListData.

TotalType[{kspace, types}, type] calcualtes the Total of the kspace data on type, where type is part of types.

TotalType[kspace, types, {type,...}] calcualtes the Total of the kspace data on each of the list type, where type is part of types.

Output is {kspace, types}.

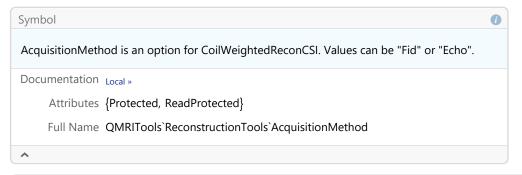
Documentation Local *

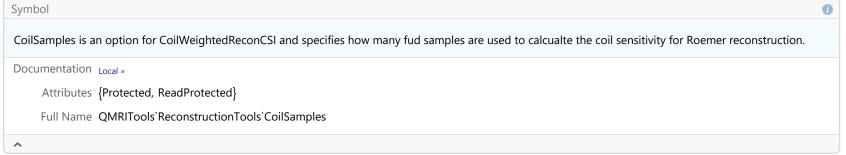
Default Definitions SyntaxInformation[TotalType] = {ArgumentsPattern → { _ , _ , _ _ ...}}}

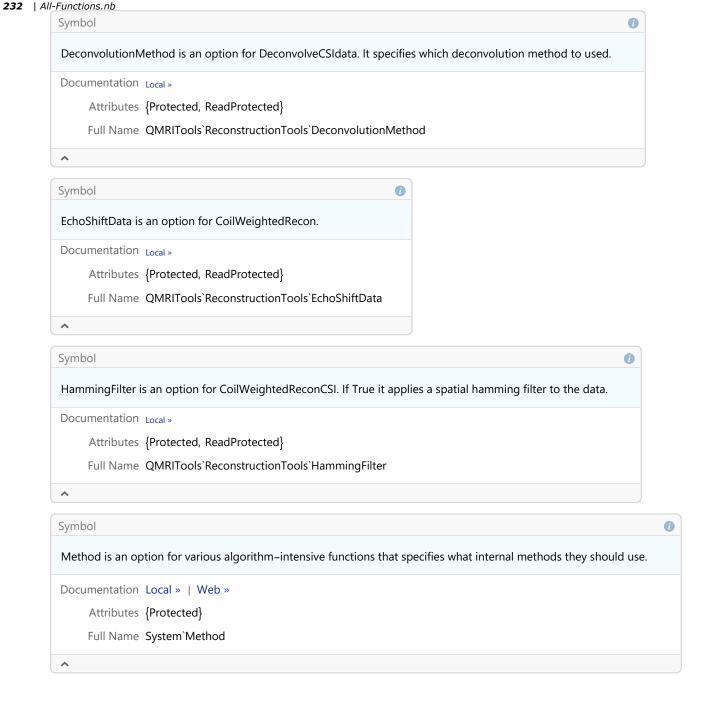
Attributes {Protected, ReadProtected}

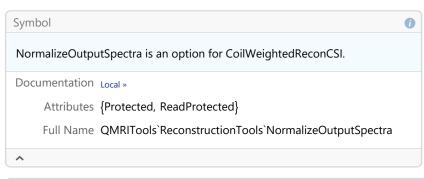
Full Name QMRITools ReconstructionTools TotalType
```

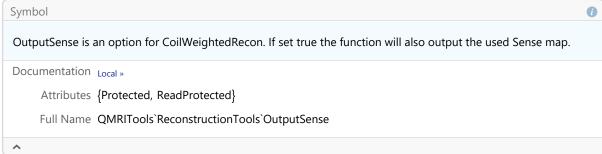
# **Options**

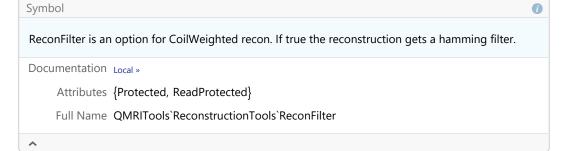


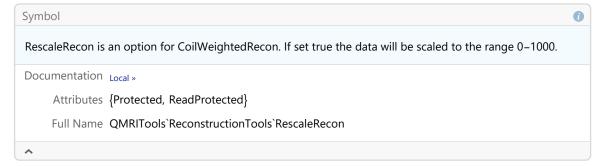


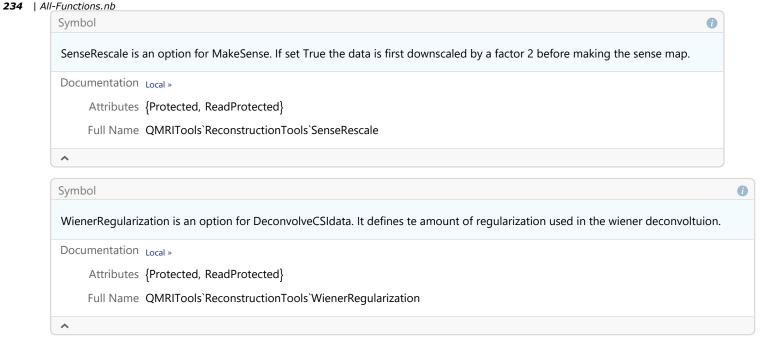






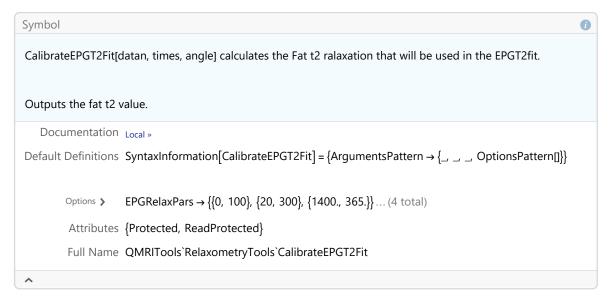






# RelaxometryTools

### **Functions**



```
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}.

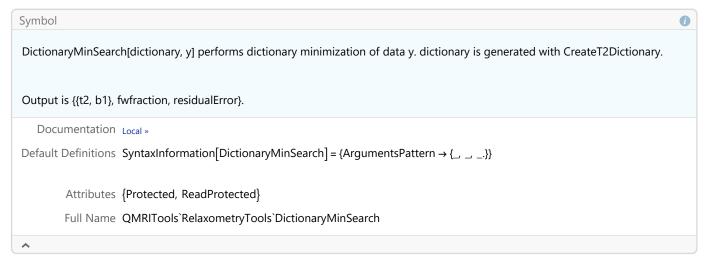
Documentation Local >>

Default Definitions SyntaxInformation[CreateT2Dictionary] = {ArgumentsPattern → { __ _ _ _ _ _ _ OptionsPattern[]}}

Options > DictB1Range → {0.5, 1.4, 0.01}... (6 total)

Attributes {Protected, ReadProtected}

Full Name QMRITools'RelaxometryTools'CreateT2Dictionary
```



Symbol

EPGSignal[{nEcho, echoSpace}, {t1, t2}, {ex\_angle,ref\_angle}, b1] generates a EPG t2 curve with stimulated echos.

t1, t2 and echoSpace are in ms, angel is in degree, b1 is between 0 and 1.

Output is the EPG Signal vector.

EPGSignal[] is based on DOI: 10.1002/jmri.24619.

Documentation Local >

Default Definitions SyntaxInformation[EPGSignal] = {ArgumentsPattern → { \_ \_ \_ \_ \_ \_ \_ \_ }}}

Attributes {Protected, ReadProtected}

Full Name QMRITools`RelaxometryTools`EPGSignal

Default Definitions SyntaxInformation[EPGT2Fit] = {ArgumentsPattern → {, , , , , OptionsPattern[]}}

Options ➤ EPGRelaxPars → {1400,, 365.} ... (17 total)

Attributes {Protected, ReadProtected}

Full Name QMRITools`RelaxometryTools`EPGT2Fit

^

NonLinearEPGFit[{vals, T2cons}, y] performs dictionary minimization of data y. vals = {{T1muscle, T1fat, T2fat}, {nEcho, echoSpace, angle}}.

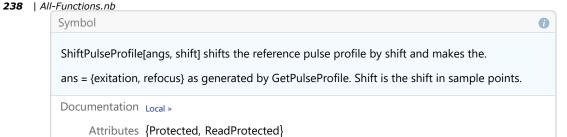
Output is {{t2, b1}, fwfraction, residualError}.

Documentation Local »

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

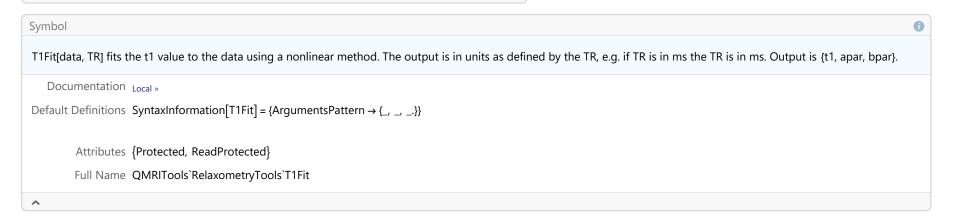
Attributes {Protected, ReadProtected}

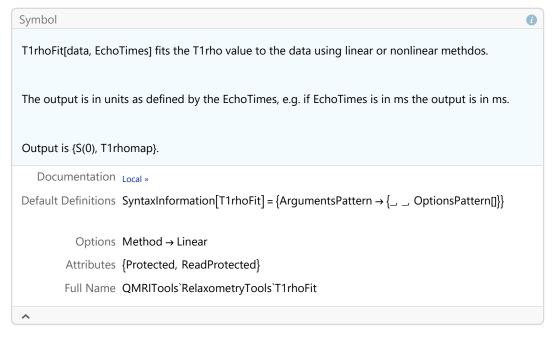
Full Name QMRITools`RelaxometryTools`NonLinearEPGFit

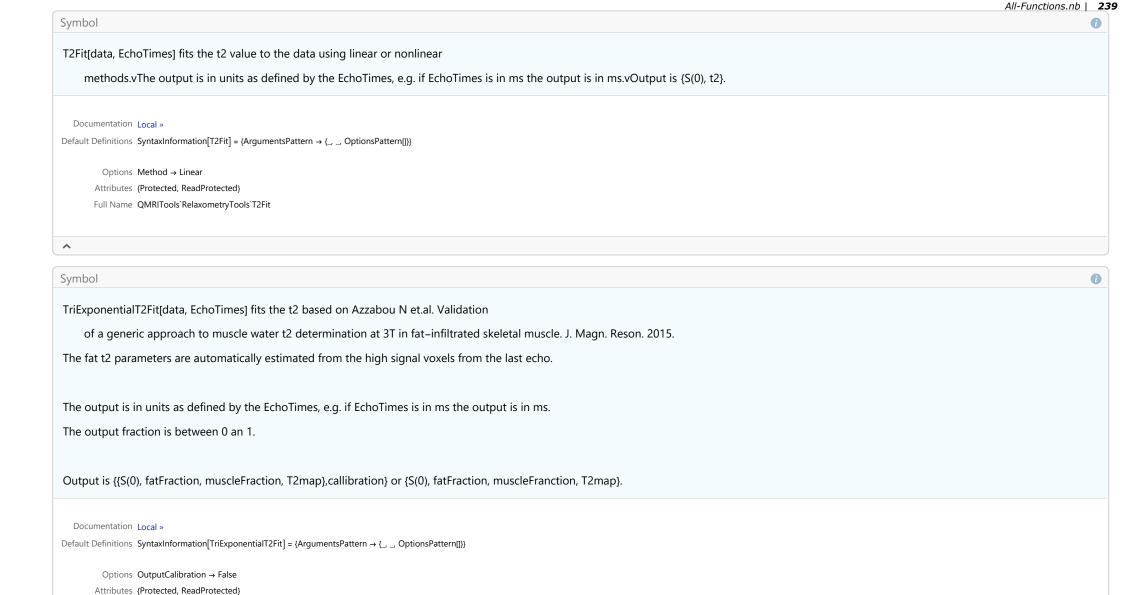


Full Name QMRITools`RelaxometryTools`ShiftPulseProfile

^

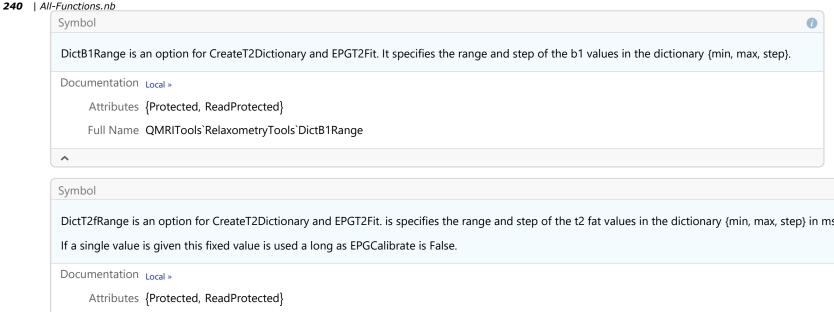


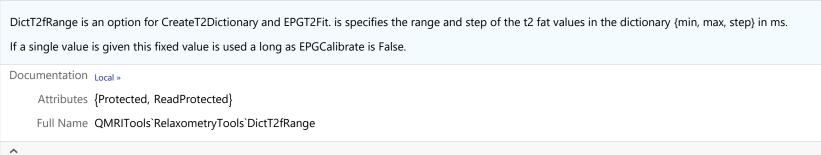




# **Options**

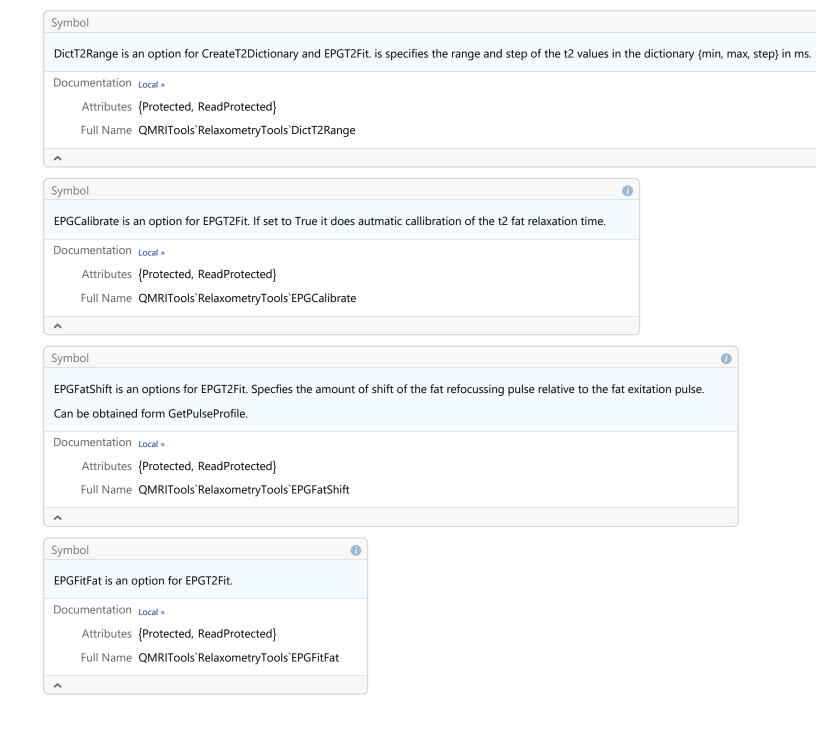
Full Name QMRITools'RelaxometryTools'TriExponentialT2Fit

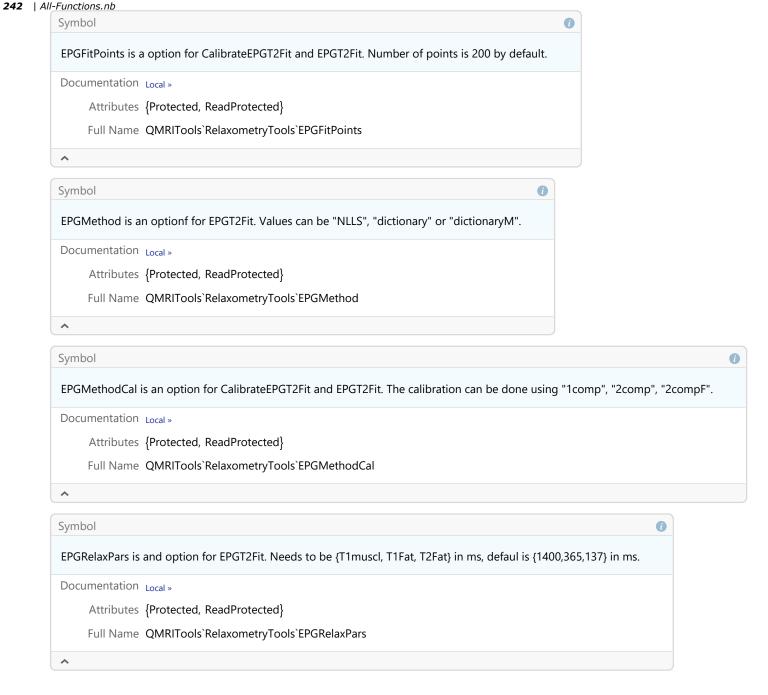


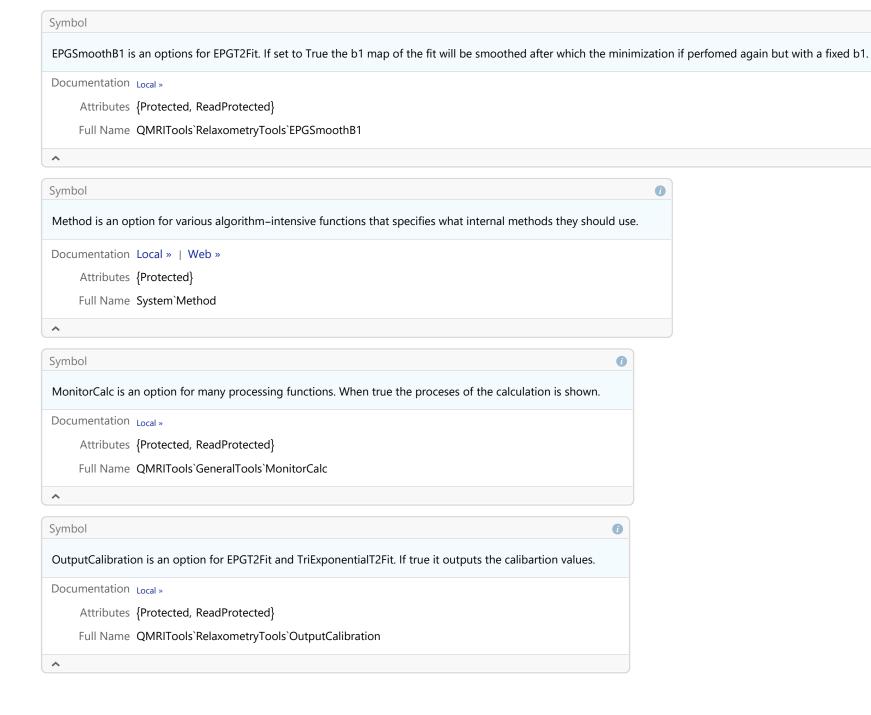


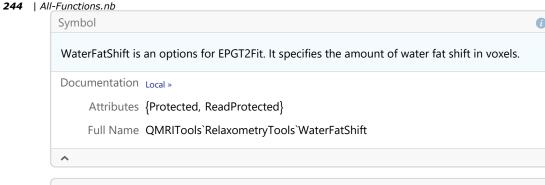


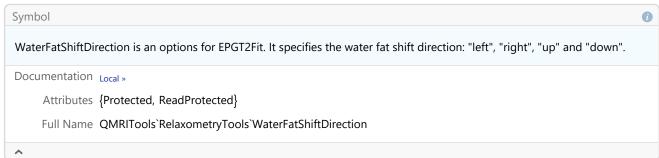






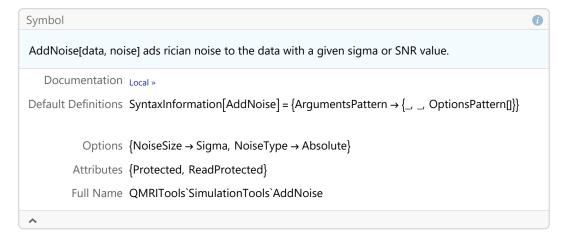


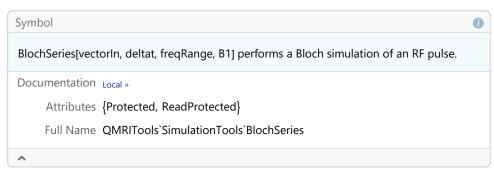


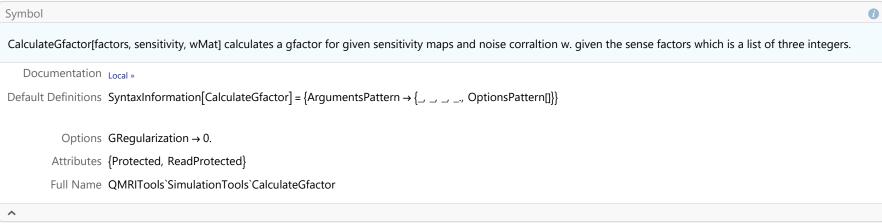


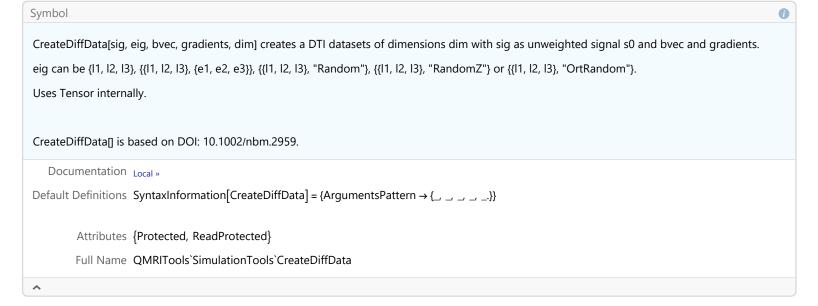
# **SimulationTools**

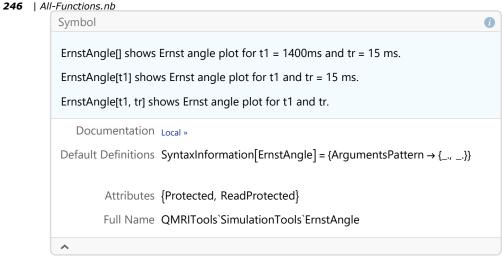
# **Functions**

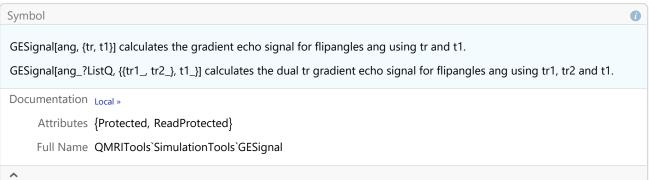












GetPulseProfile[excitation, refocus] gives the pusl angle profiles for the exitation and refocussing pulses.

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[]}}

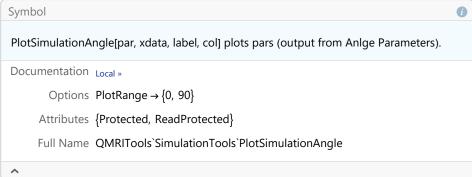
Options > MagnetizationVector → {0, 0, 1} ... (4 total)

Attributes {Protected, ReadProtected}

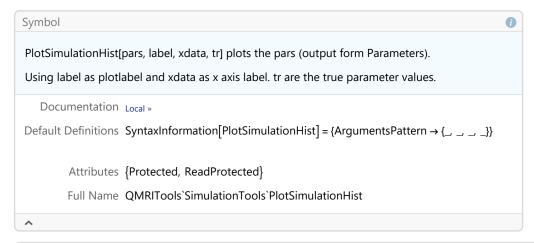
Full Name QMRITools`SimulationTools`GetPulseProfile

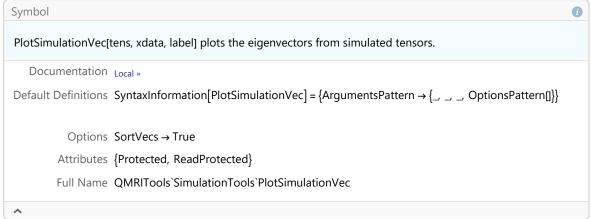
# GfactorSimulation[sensitivity, cov, {dir,sense}] calculates the gfactormaps for given sensitivity maps and noise corraltion cov in one direction. The sensefactors are a list of integers in a given direction: "LR", "FH", or "AP". GfactorSimulation[sensitivity, cov, {dir1,sense1}, {dir2,sense2}] calculates the gfactormaps for given sensitivity maps and noise corraltion w in two directions. Documentation Local >> Default Definitions SyntaxInformation[GfactorSimulation] = {ArgumentsPattern → {\_, \_, \_, \_, \_, OptionsPattern[]}} Options {GRegularization → 0., GOutput → Grid} Attributes {Protected, ReadProtected} Full Name QMRITools'SimulationTools'GfactorSimulation

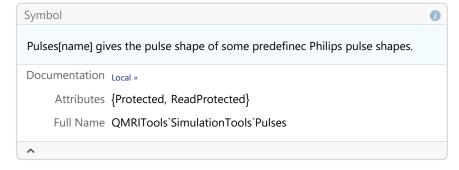
248 | All-Functions.nb Symbol PlotSimulation[pars, xval, true, label, color] plots the pars (output form Parameters). Using label as PlotLabel and xval as x axis Thics. tr are the true parameter values. color are the color used for the plot. Documentation Local » Default Definitions SyntaxInformation[PlotSimulation] = {ArgumentsPattern → {\_, \_, \_, \_, \_, OptionsPattern[]}} Options PlotRange  $\rightarrow \{\{0, 3\}, \{0, 3\}, \{0, 3\}, \{0, 3\}, \{0, 1\}\}$ Attributes {Protected, ReadProtected} Full Name QMRITools`SimulationTools`PlotSimulation  $\wedge$ Symbol 0 PlotSimulationAngle[par, xdata, label, col] plots pars (output from Anlge Parameters). Documentation Local »

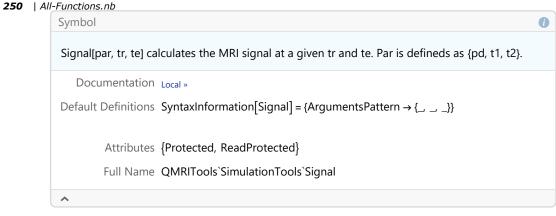


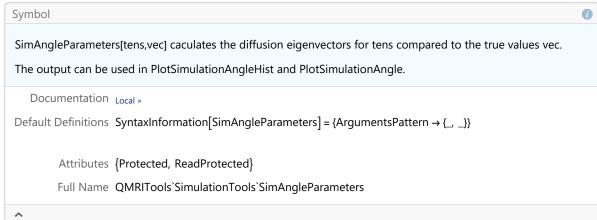
Symbol 0 PlotSimulationAngleHist[pars, label, xdata] plots pars (output from Anlge Parameters). Documentation Local » Default Definitions SyntaxInformation[PlotSimulationAngleHist] = {ArgumentsPattern → {\_, \_, \_, \_, \_, \_, OptionsPattern[]}} Attributes {Protected, ReadProtected} Full Name QMRITools`SimulationTools`PlotSimulationAngleHist  $\wedge$ 

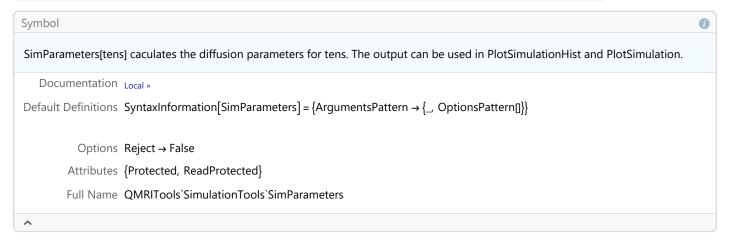


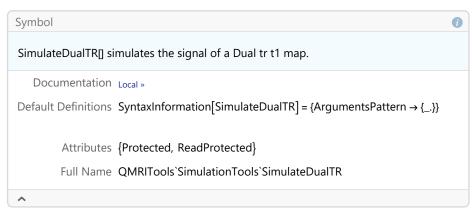


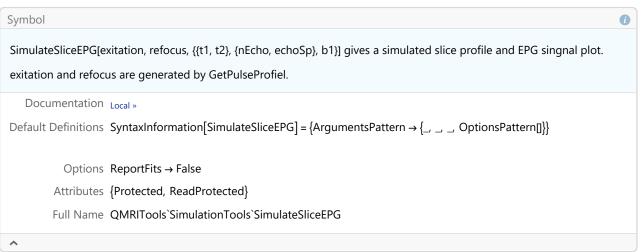












Tensor[{|1, |2, |3}| creates a diffuison tensor with vectors {{0,0,1},{0,1,0},{1,0,0}} and eigenvalues {|1, |2, |3}.

Tensor[{|1, |2, |3}, {e1, e2, e3}| creates a diffuison tensor with vectors {e1, e2, e3} and eigenvalues {|1, |2, |3}.

Tensor[{|1, |2, |3}, "Random"] creates a diffuison tensor with random orthogonal eigenvectors {e1, e2, e2} and eigenvalues {|1, |2, |3}.

Tensor[{|1, |2, |3}, "RandomZ"] creates a diffuison tensor with random orthogonal eigenvectors {{1,0,0}, e2, e3} with random eigenvectors and eigenvalues {|1, |2, |3}.

Tensor[{|1, |2, |3}, "OrtRandom"] creates a diffuison tensor with random orthogonal eigenvectors {{1,0,0},{0,1,0},{0,0,1}} and eigenvalues {|1, |2, |3}.

Tensor[] is based on DOI: 10.1002/nbm.2959.

Documentation Local \*\*

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

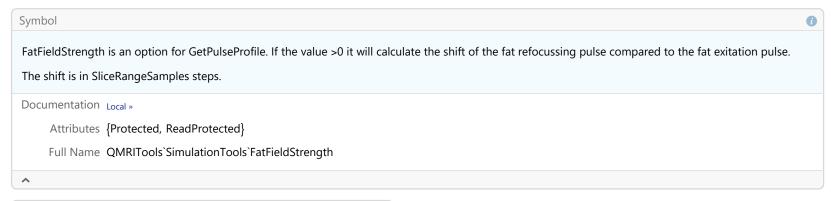
Options TensOutput → Vector

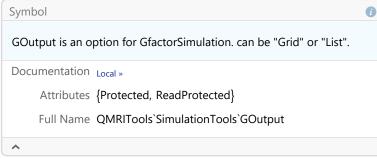
Attributes {Protected, ReadProtected}}

Full Name QMRITools'SimulationTools'Tensor

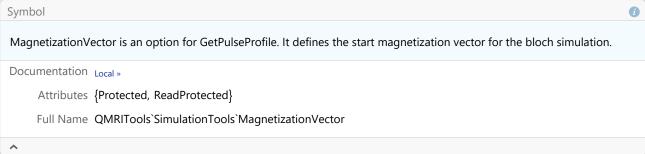
### **Options**

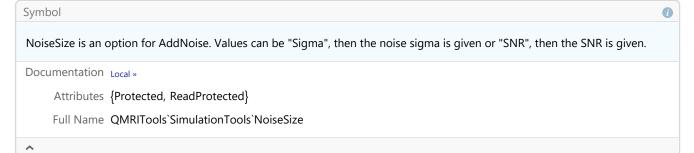
 $\wedge$ 

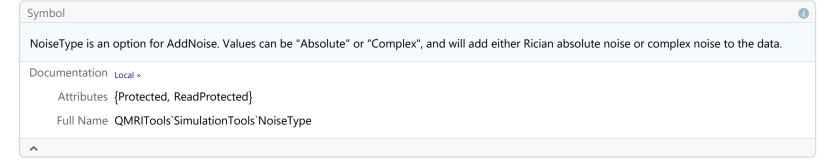


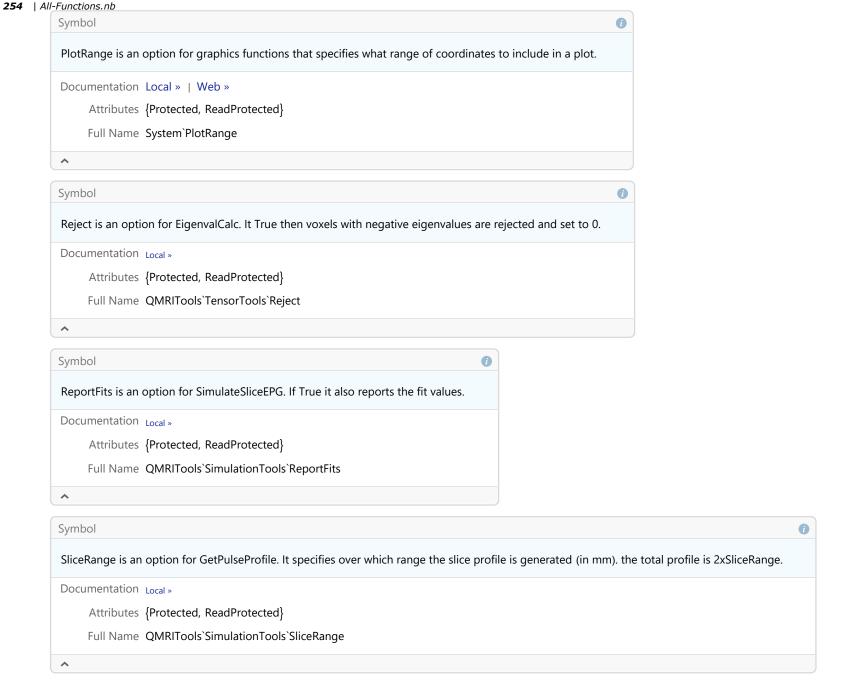


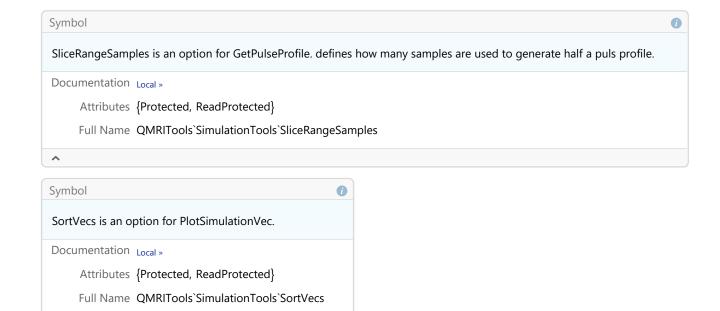


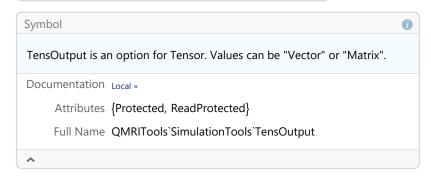






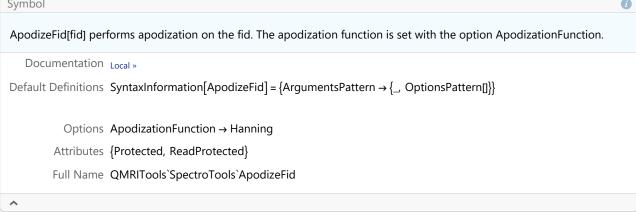






# **SpectroTools**

**Functions** 



Symbol ApodizePadEcho[echo] performs apodization on the echo and pads the echo with zeros to increase its length. Documentation Local » Default Definitions SyntaxInformation[ApodizePadEcho] = {ArgumentsPattern → {\_, OptionsPattern[]}} Options {ApodizationFunction  $\rightarrow$  Hanning, PaddingFactor  $\rightarrow$  2} Attributes {Protected, ReadProtected} Full Name QMRITools`SpectroTools`ApodizePadEcho  $\wedge$ 

```
ApodizePadFid[fid] performs apodization on the fid and pads the fid with zeros to increase its length.

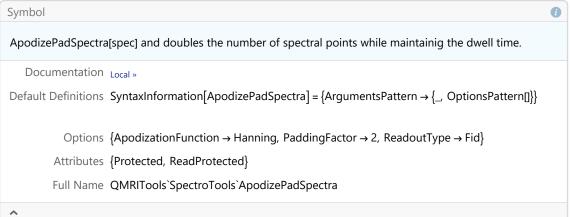
Documentation Local »

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

Options {ApodizationFunction → Hanning, PaddingFactor → 2}

Attributes {Protected, ReadProtected}

Full Name QMRITools`SpectroTools`ApodizePadFid
```



```
ApodizeSpectra[spec] performs apodization of the spectra. The apodization function is set with the option ApodizationFunction.

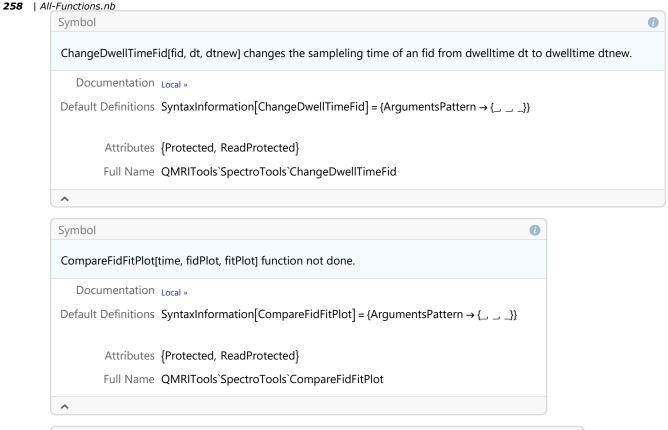
Documentation Local >>

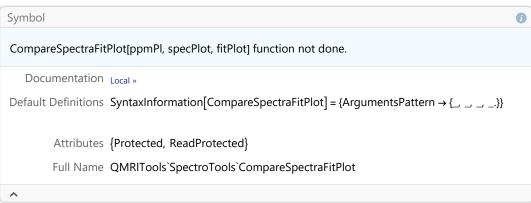
Default Definitions SyntaxInformation[ApodizeSpectra] = {ArgumentsPattern → {_, OptionsPattern[]}}

Options {ApodizationFunction → Hanning, ReadoutType → Fid}

Attributes {Protected, ReadProtected}

Full Name QMRITools`SpectroTools`ApodizeSpectra
```





CorrectTEFid[fid, dw, te] corrects the fid for 1st order phase by extrapolating the missing FID samples in the TE using Henkel matrix SVD analsis.

CorrectTEFid[fid, dw, te, gyro, ppmRan] corrects the fid for 1st order phase by extrapolating

the missing FID samples in the TE using Henkel matrix SVD analsis. Only the part of the spectra in the ppmRan is used for optimization.

Documentation Local »

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

Attributes {Protected, ReadProtected}

Full Name QMRITools`SpectroTools`CorrectTEFid

^

#### Symbol



CorrectTESpec[spectra, dw, te, gyro, ppmRan] corrects the spectra for 1st order phase by extrapolating

the missing FID samples in the TE using Henkel matrix SVD analsis. Only the part of the spectra in the ppmRan is used for optimization.

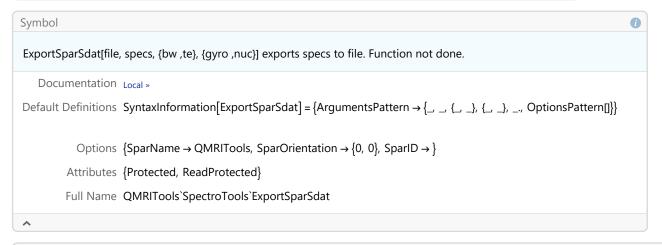
Documentation Local »

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

Attributes {Protected, ReadProtected}

Full Name QMRITools`SpectroTools`CorrectTESpec

```
260 | All-Functions.nb
           Symbol
           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 \rightarrow 7, SpectraNucleus \rightarrow 31P}
                    Attributes {Protected, ReadProtected}
                    Full Name QMRITools`SpectroTools`CSIInterface
           ^
```



```
Symbol
FindSpectraPpmShift[spectra, {dw, gyro}, peaks] finds the ppm value that aligns the spectra with the given peak positions peaks wich is a list of ppm values.
FindSpectraPpmShift[spectra, {dw, gyro}, {peaks, amps}] finds the ppm value
    that aligns the spectra with the given peak positions peaks wich is a list of ppm values and amps are ther relative amplitudes.
FindSpectraPpmShift[spectra, {dw, gyro}, specTar] finds the ppm value that aligns the spectra with the given target specTar.
Documentation Local »
    Attributes {Protected, ReadProtected}
    Full Name QMRITools`SpectroTools`FindSpectraPpmShift
```

```
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.

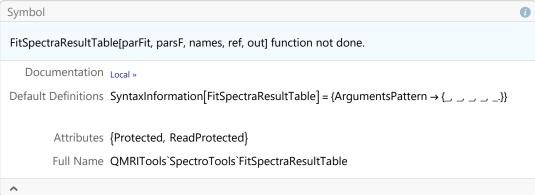
Documentation Local >

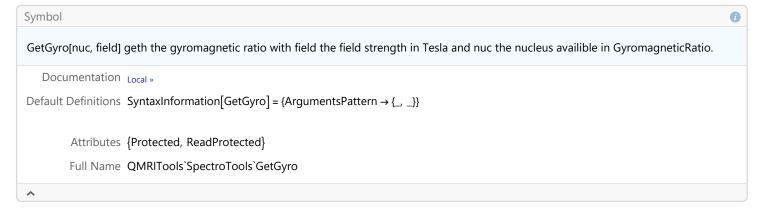
Default Definitions SyntaxInformation[FitSpectra] = {ArgumentsPattern → { __ _ _ _ _ _ _ _ OptionsPattern[]}}

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

Attributes {Protected, ReadProtected}

Full Name QMRITools SpectroTools FitSpectra
```





0

GetPpmRange[spec, {dt, field, nuc}] get the ppm values of the spec where dt is the well time in ms, field the field strength in Tesla and nuc the nucleus availible in GyromagneticRatio.

GetPpmRange[spec, dt, field, nuc] get the ppm values of the spec.

GetPpmRange[spec, dt, gyro] get the ppm values of the spec.

Documentation Local »

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

Attributes {Protected, ReadProtected}

Full Name QMRITools`SpectroTools`GetPpmRange

Symbol

 $\wedge$ 

 $\wedge$ 

GetSpectraBasisFunctions[{met1, ..., metn}] generates a list of spectra baisis functions with names met1 to metn. The names are strings and are the metabolites availible in GetSpinSystem.

GetSpectraBasisFunctions[{{props1}, ..., {propsn}}] generates a list of spectra baisis functions with properties prop1 to propn. The properties are those specified in MakeSpinSystem.

GetSpectraBasisFunctions[inp, split] generates a list of spectra basisfunctions. Each metabolite name present in the list split wil be split in individual spectra per peak.

Documentation Local »

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

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

Attributes {Protected, ReadProtected}

Full Name QMRITools`SpectroTools`GetSpectraBasisFunctions

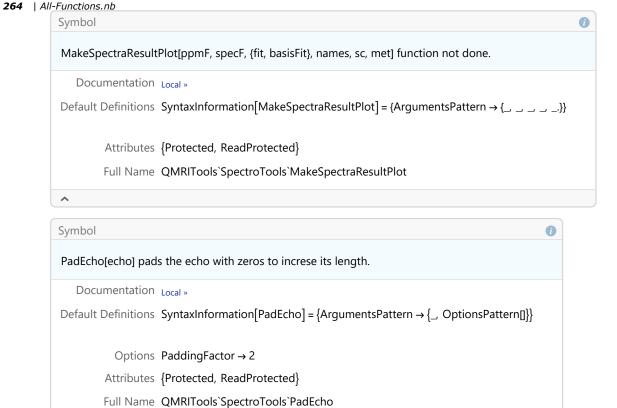
ImportSparSdat[fspar, fsdat] imports sfpar and fsdata file. Function not done.

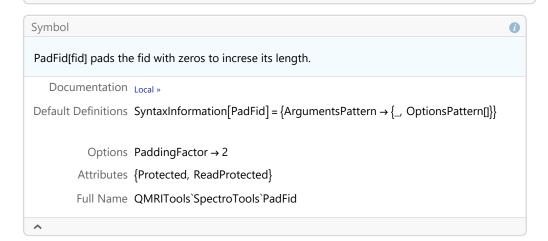
Documentation Local »

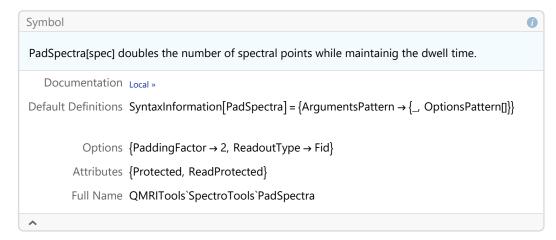
Default Definitions SyntaxInformation[ImportSparSdat] = {ArgumentsPattern → {\_, \_}}}

Attributes {Protected, ReadProtected}

Full Name QMRITools`SpectroTools`ImportSparSdat







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.

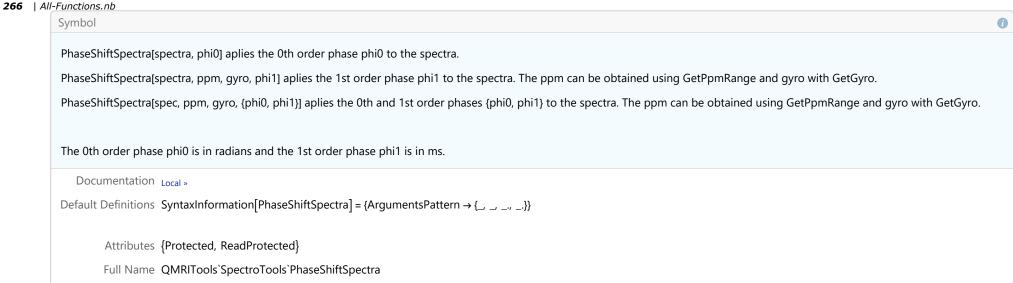
Documentation Local »

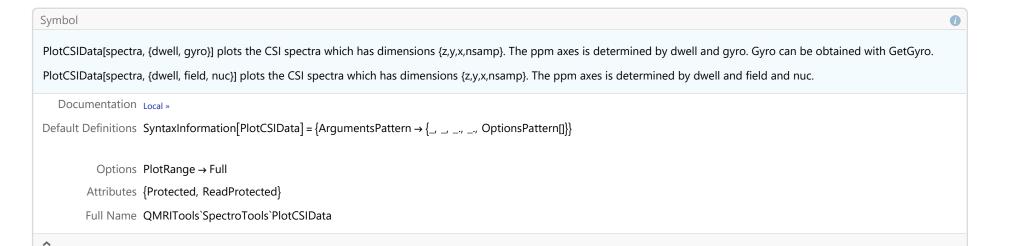
Default Definitions SyntaxInformation[PhaseCorrectSpectra] = {ArgumentsPattern → {\_, \_, \_, \_, \_, \_, \_,}}

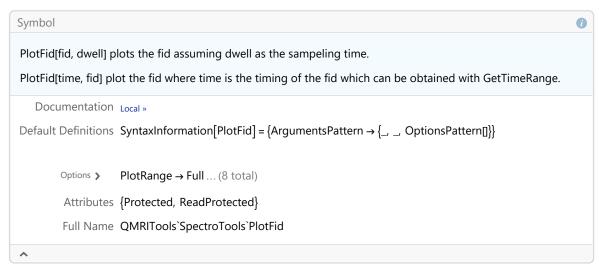
Attributes {Protected, ReadProtected}

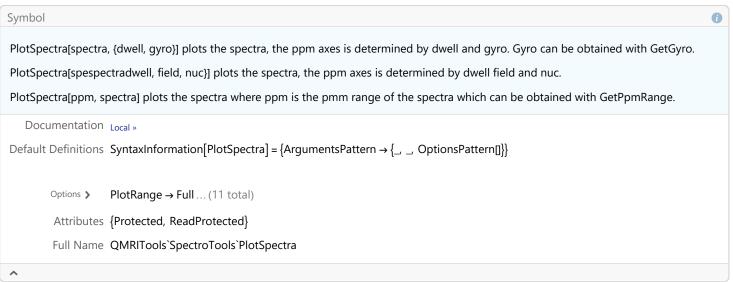
Full Name QMRITools`SpectroTools`PhaseCorrectSpectra

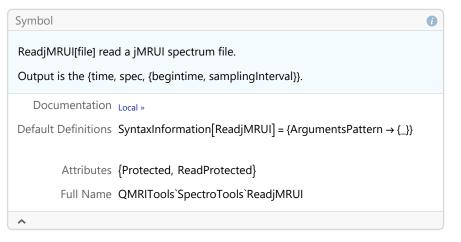
^

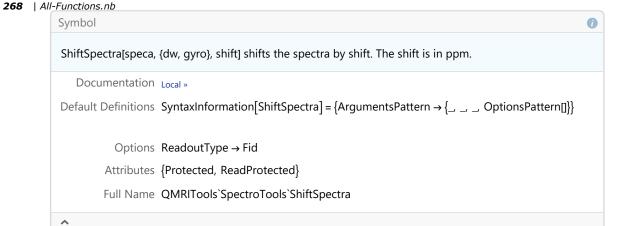


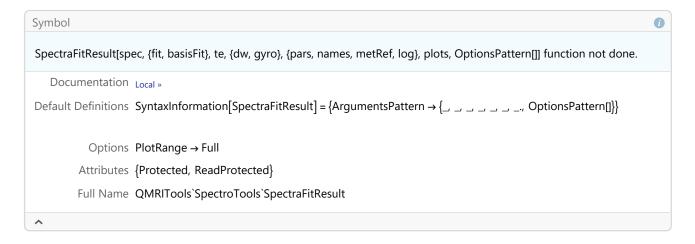












TimeShiftEcho[fid, time, gam] aplies a linebroadening with linewidth gam and a Voigt lineshape to the fid. The time can be obtained using GetTimeRange.

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  $\rightarrow$  {\_\_ \_\_\_, \_\_.}}

Attributes {Protected, ReadProtected}

Full Name QMRITools`SpectroTools`TimeShiftEcho

^

#### Symbol



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.

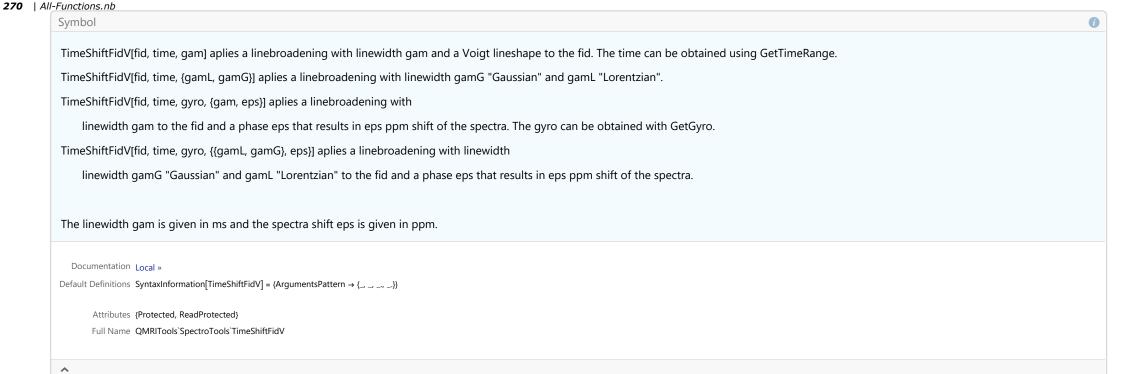
The linewidth gam is given in ms and the spectra shift eps is given in ppm.

Documentation Local »

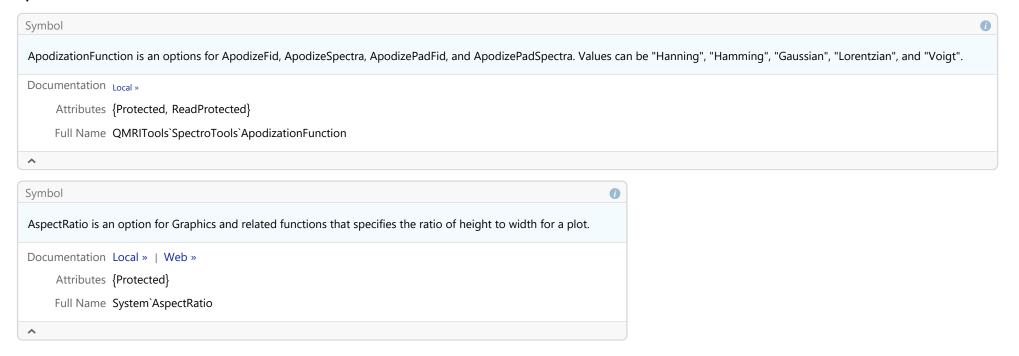
Default Definitions SyntaxInformation[TimeShiftFid] = {ArgumentsPattern → {\_, \_, \_, \_.}}

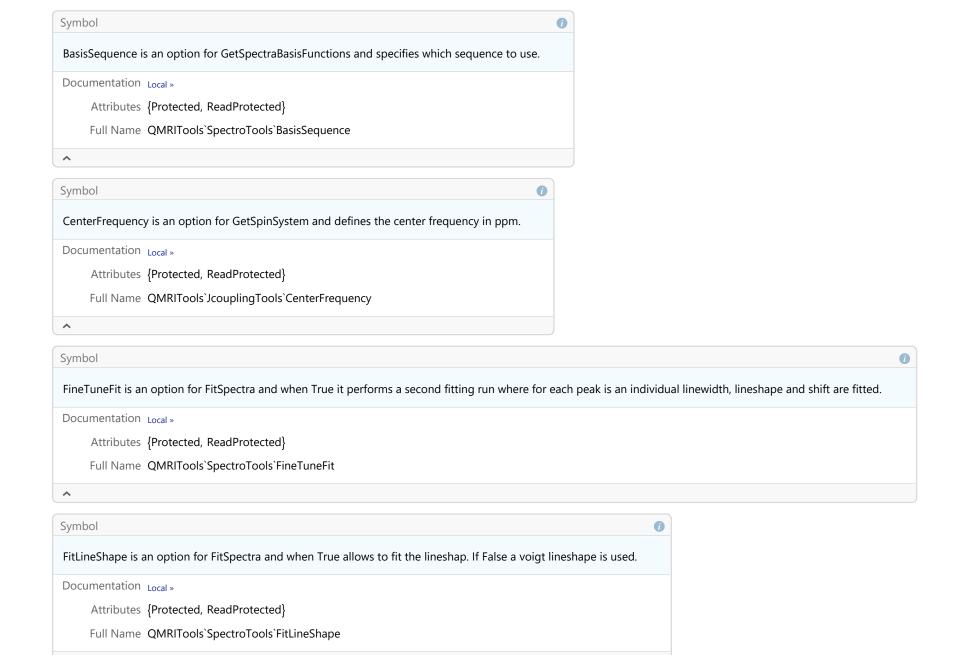
Attributes {Protected, ReadProtected}

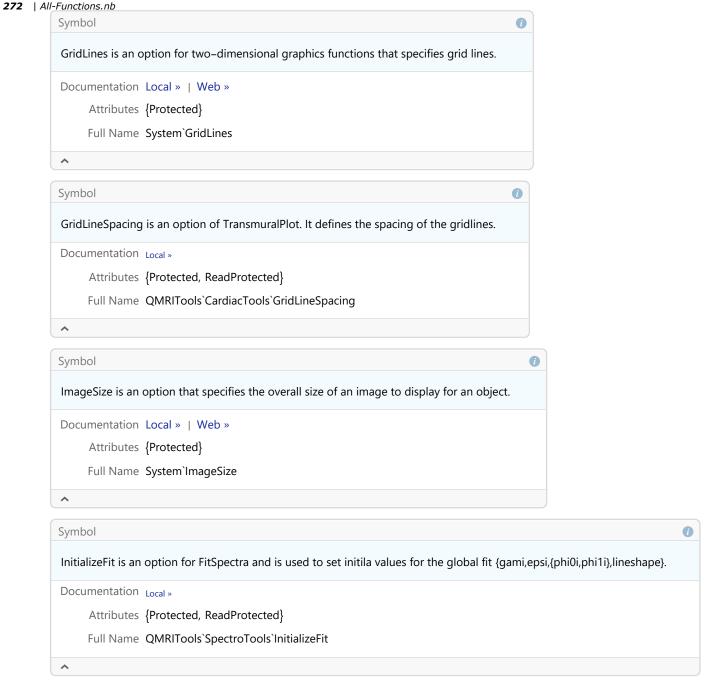
Full Name QMRITools`SpectroTools`TimeShiftFid

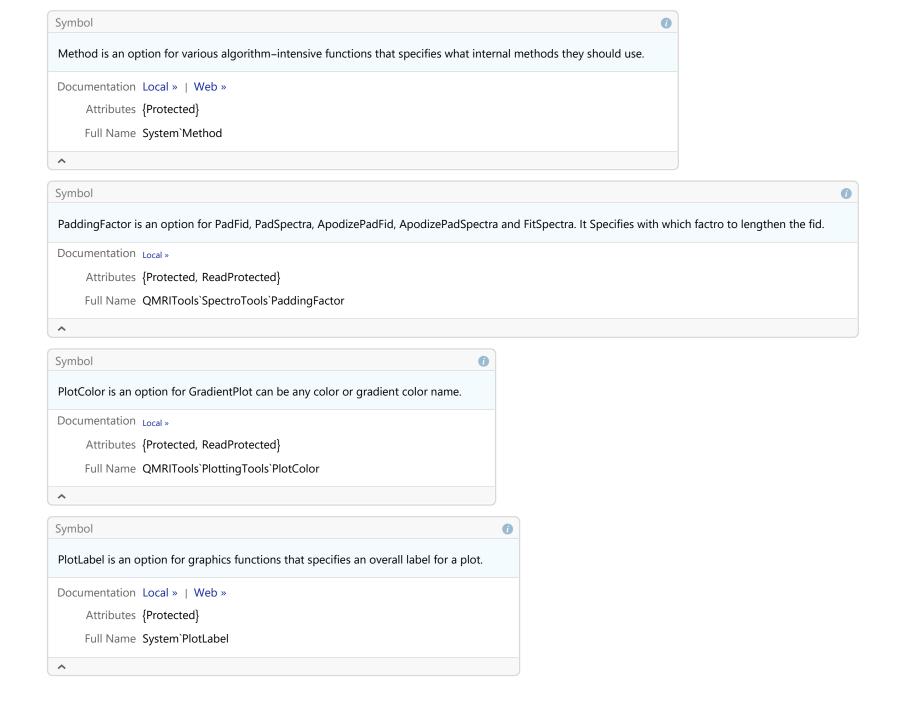


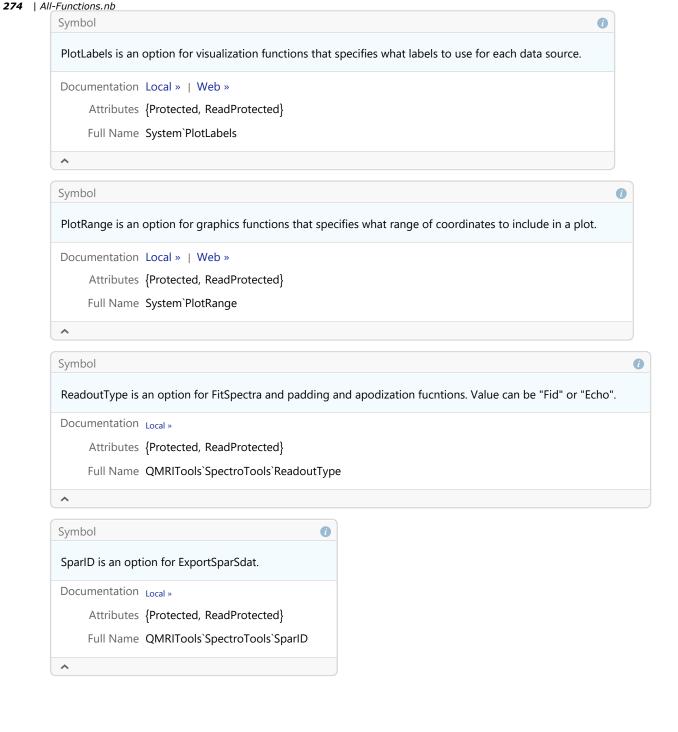
## **Options**

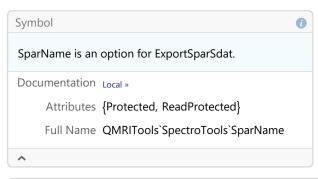


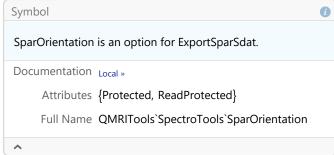


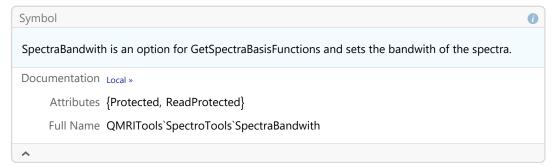


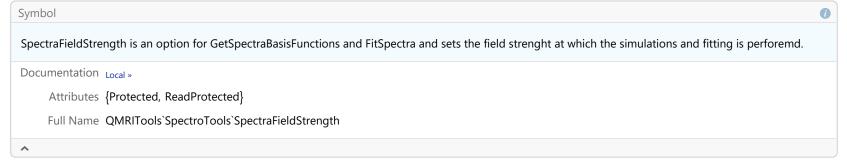


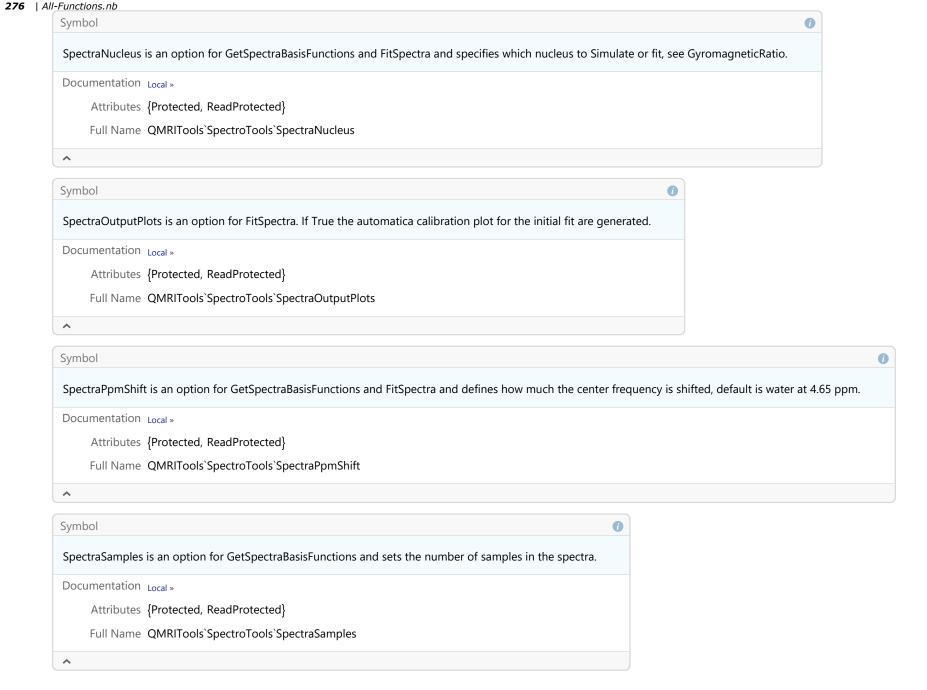


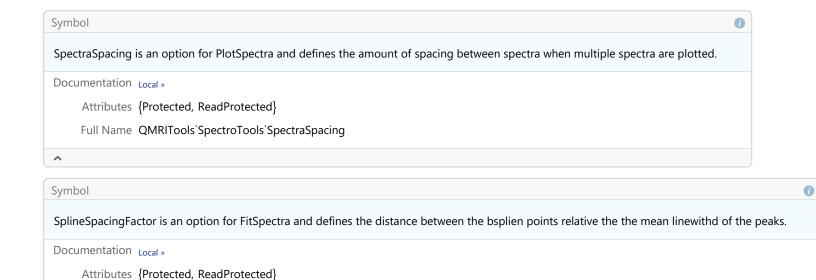






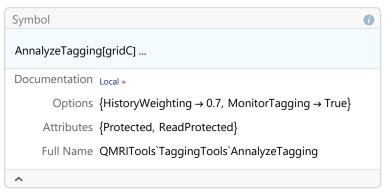






# TaggingTools

## **Functions**



Full Name QMRITools`SpectroTools`SplineSpacingFactor

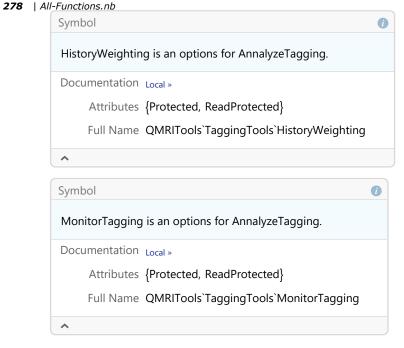
```
CalculateDispacementParameters[{motx, moty}, mask] ...

Documentation Local »

Attributes {Protected, ReadProtected}

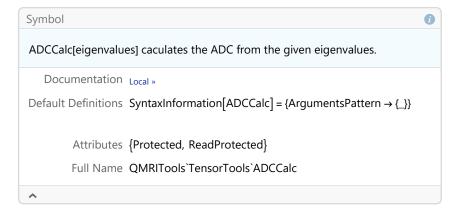
Full Name QMRITools`TaggingTools`CalculateDispacementParameters
```

# **Options**

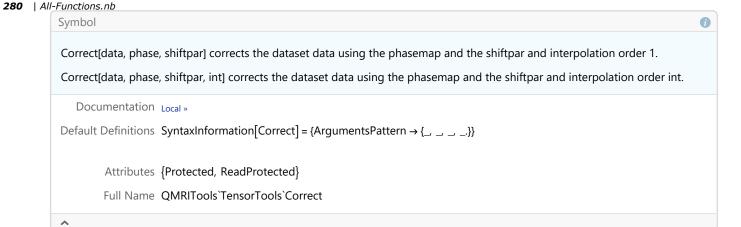


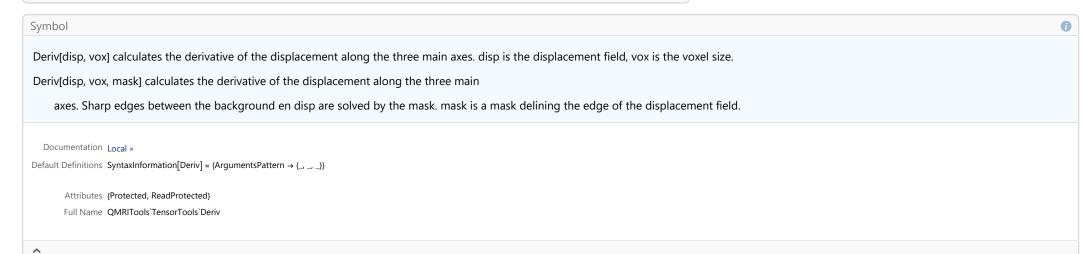
# **TensorTools**

## **Functions**



```
Symbol
AngleCalc[data, vector] calculates the angel between the vector and the data. Data shoud be an array of dimensions {xxx,3}.
   Documentation Local »
Default Definitions SyntaxInformation[AngleCalc] = {ArgumentsPattern → {_, _, OptionsPattern[]}}
           Options Distribution \rightarrow 0–180
         Attributes {Protected, ReadProtected}
         Full Name QMRITools`TensorTools`AngleCalc
^
Symbol
AngleMap[data] calculates the zennith and azimuth angles of a 3D dataset (z,x,y,3) containing vectors relative to the slice direction.
   Documentation Local »
Default Definitions SyntaxInformation[AngleMap] = {ArgumentsPattern \rightarrow {_}}
         Attributes {Protected, ReadProtected}
         Full Name QMRITools`TensorTools`AngleMap
^
Symbol
ConcatenateDiffusionData[{{data1, .., dataN}, {grad1, .., gradN}, {bval, .., bvalN}, {vox, .., voxN}}] concatenates the diffusion data sets.
ConcatenateDiffusionData[{data1, .., dataN}, {grad1, .., gradN}, {bval, .., bvalN}, {vox, .., voxN}] concatenates the diffusion data sets.
   Documentation Local »
Default Definitions SyntaxInformation[ConcatenateDiffusionData] = {ArgumentsPattern \rightarrow {________}}
         Attributes {Protected, ReadProtected}
         Full Name OMRITools`TensorTools`ConcatenateDiffusionData
\wedge
```





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.

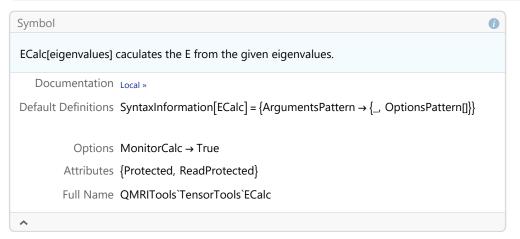
Documentation Local \*\*

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

Options {NormalizeSignal → True, UseMask → True}

Attributes {Protected, ReadProtected}

Full Name QMRITools'TensorTools'DriftCorrect



EigensysCalc[tensor] caculates the eigensystem for the given tensor.

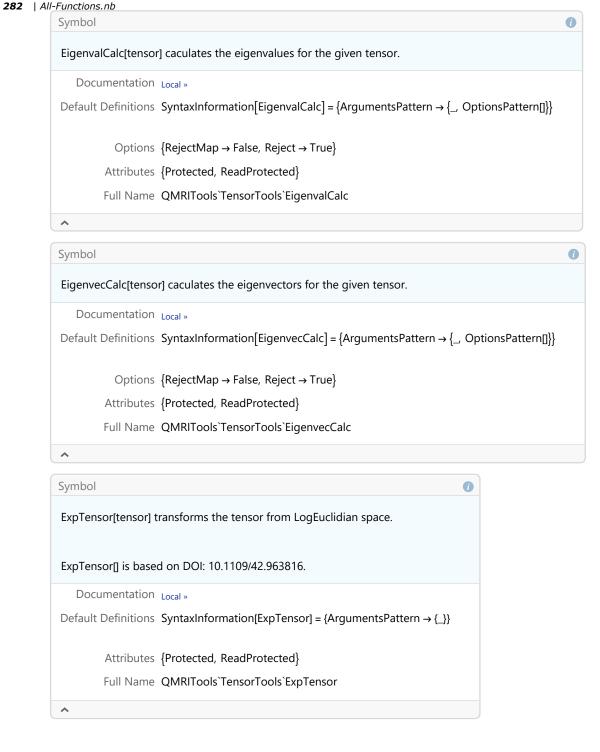
Documentation Local »

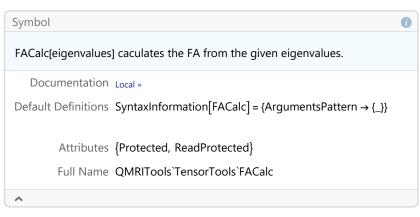
Default Definitions SyntaxInformation[EigensysCalc] = {ArgumentsPattern → {\_, OptionsPattern[]}}

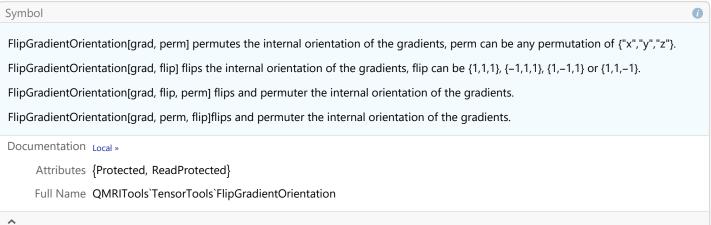
Options {RejectMap → False, Reject → True}

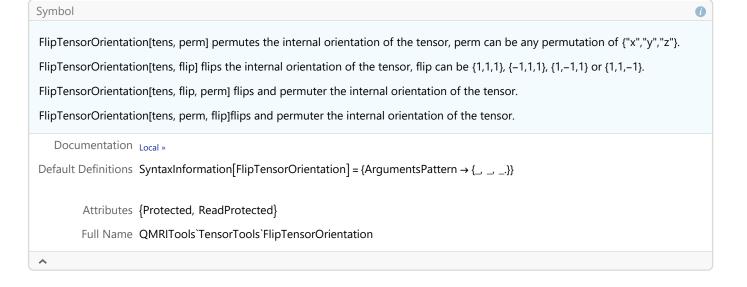
Attributes {Protected, ReadProtected}

Full Name QMRITools`TensorTools`EigensysCalc

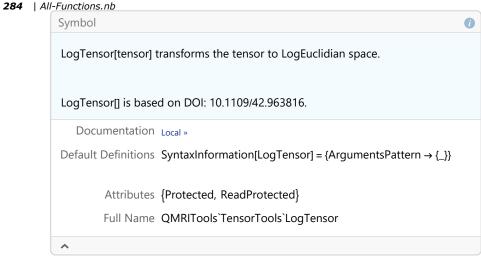


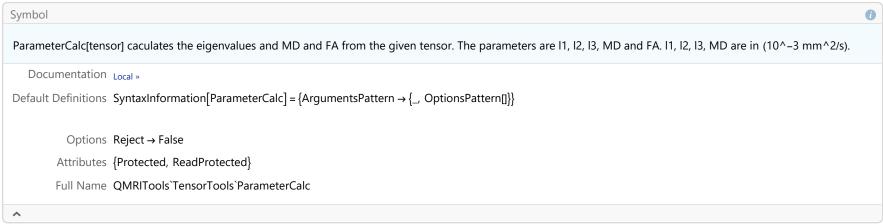














ResidualCalc[dti,{tensor,s0},gradients,bvector] calculates the tensor residuals for the given dataset.

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].

0

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 QMRITools`TensorTools`ResidualCalc

^

#### Symbol



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

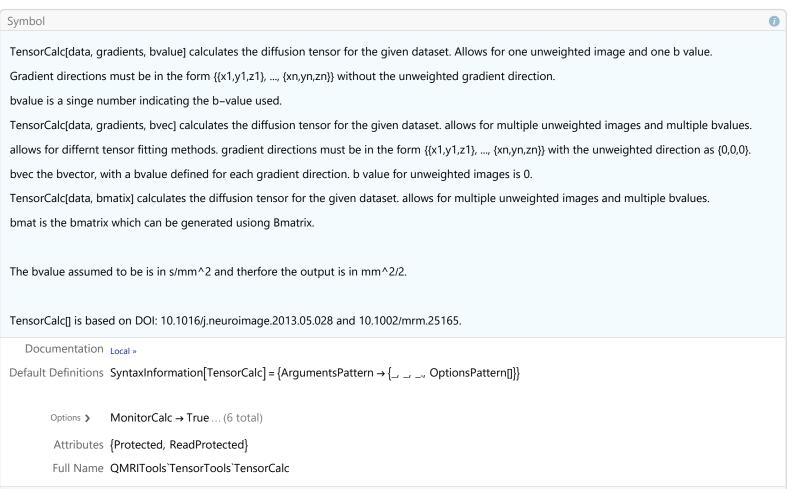
Attributes {Protected, ReadProtected}

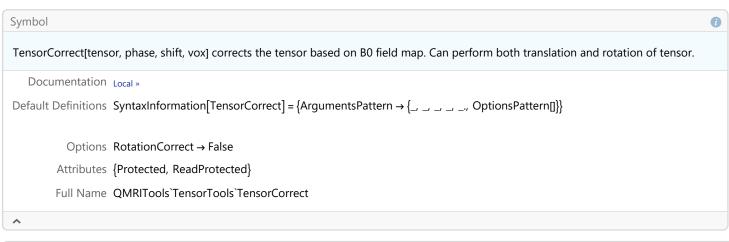
Full Name QMRITools`TensorTools`SigmaCalc

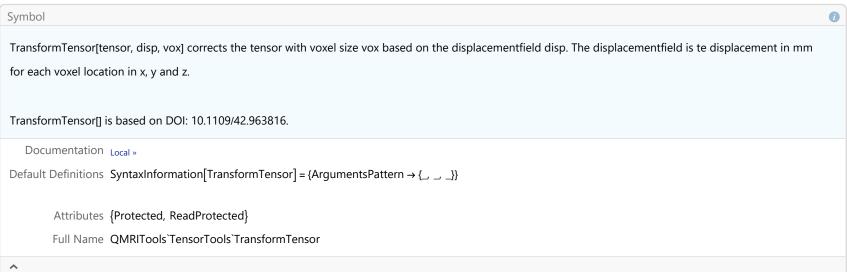
^

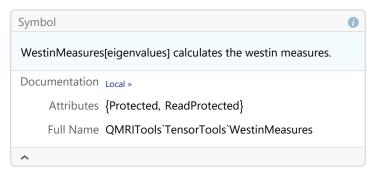
 $\wedge$ 

286 | All-Functions.nb Symbol SortDiffusionData[data, grad, bval] sorts the diffusion datasets grad and bval for magnitude of bvalue. Documentation Local » Default Definitions SyntaxInformation[SortDiffusionData] = {ArgumentsPattern  $\rightarrow \{\_, \_, \_\}$ } Attributes {Protected, ReadProtected} Full Name QMRITools`TensorTools`SortDiffusionData ^

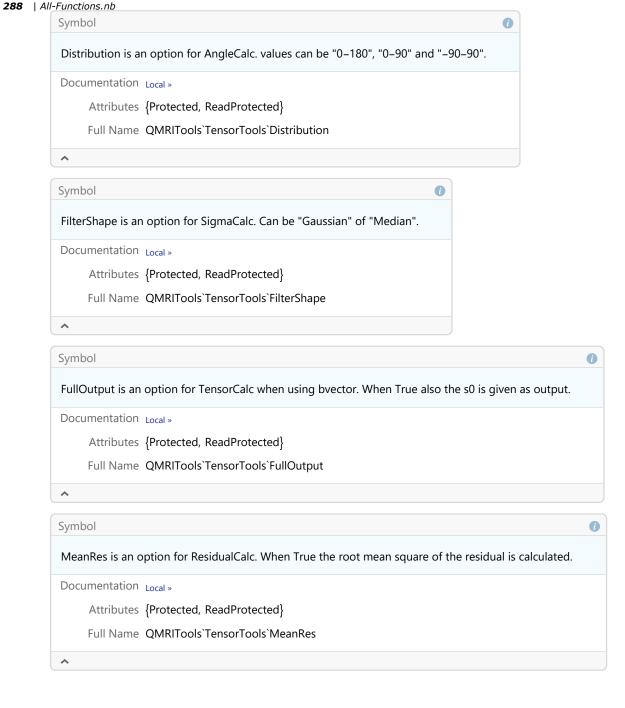


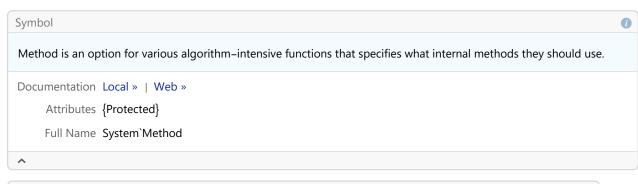


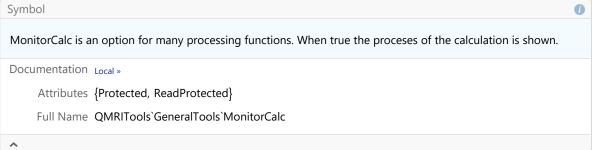




**Options** 









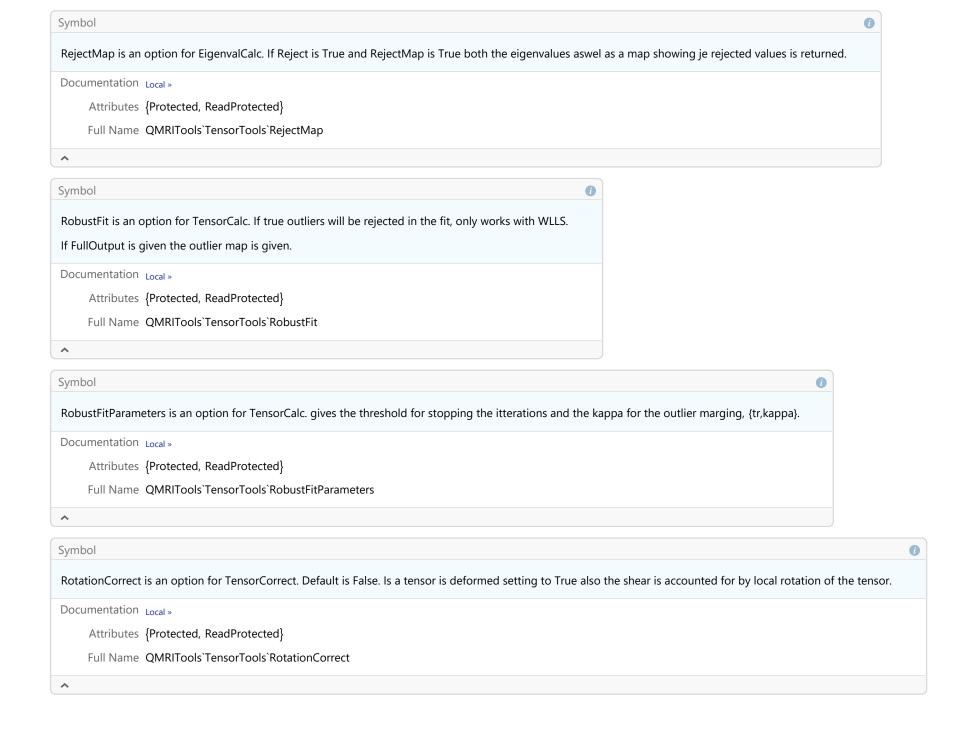
```
Reject is an option for EigenvalCalc. It True then voxels with negative eigenvalues are rejected and set to 0.

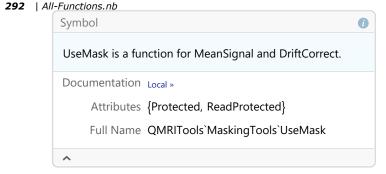
Documentation Local »

Attributes {Protected, ReadProtected}

Full Name QMRITools`TensorTools`Reject
```

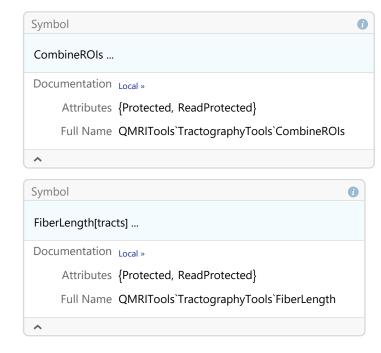
^





# ${\tt TractographyTools}$

### **Functions**



FiberTractography[tensor, vox] performs fibertractography on the tensor with voxels dimensions vox.

FiberTractography[tensor, vox, {par, {min, max}}] performs fibertractography on the tensor with voxels dimensions vox with additional stoppin criteria par, where tracts are only generated between values of par min and max.

FiberTractography[tensor, vox, {{par, {min, max}}, ...}] performs fibertractography on the tensor with voxels dimensions vox with multiple additional stopping criteria.

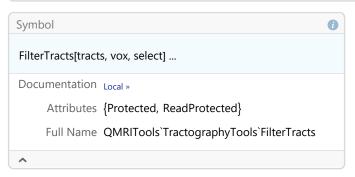
Documentation Local >>

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

Options > FiberLengthRange → {10, 200}....(10 total)

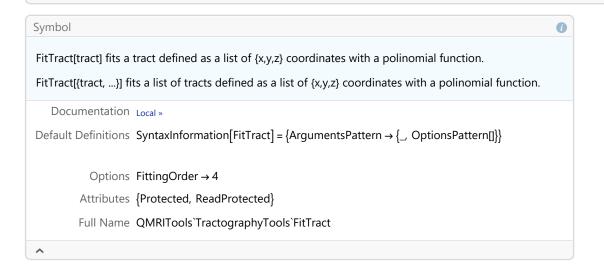
Attributes {Protected, ReadProtected}

Full Name QMRITools TractographyTools FiberTractography

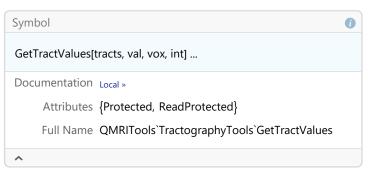


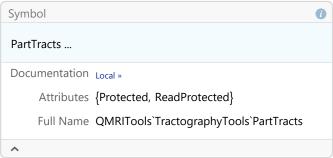
 $\wedge$ 

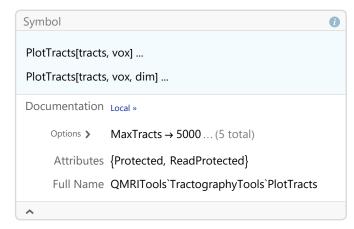
294 | All-Functions.nb Symbol FindTensorPermutation[tensor, vox] performs tractography for all tensor permutations and gives back the one that has the longest tracts. FindTensorPermutation[tensor, vox, {par, {min, max}}] same but with additional stoppin criteria par, where tracts are only generated between 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[]}} FiberLengthRange  $\rightarrow$  {10, 200}...(7 total) Options > Attributes {Protected, ReadProtected}

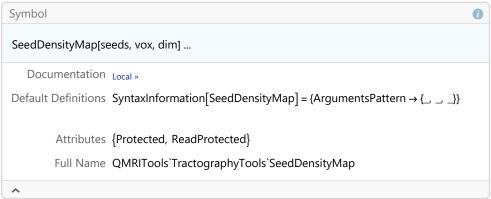


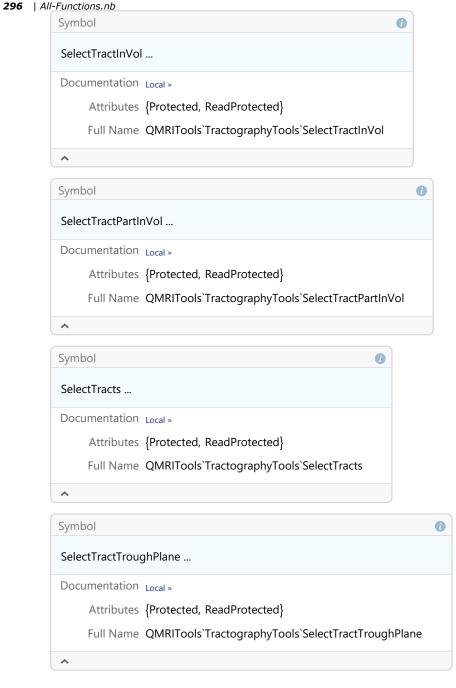
Full Name QMRITools`TractographyTools`FindTensorPermutation

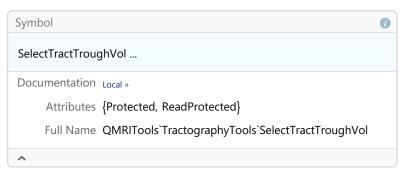


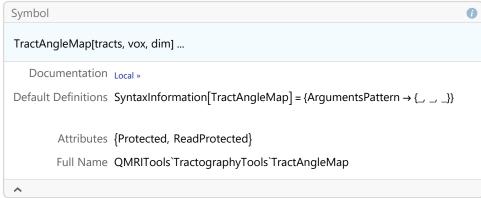


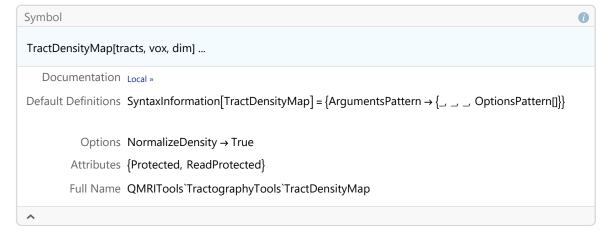


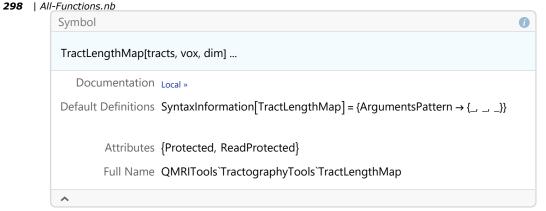


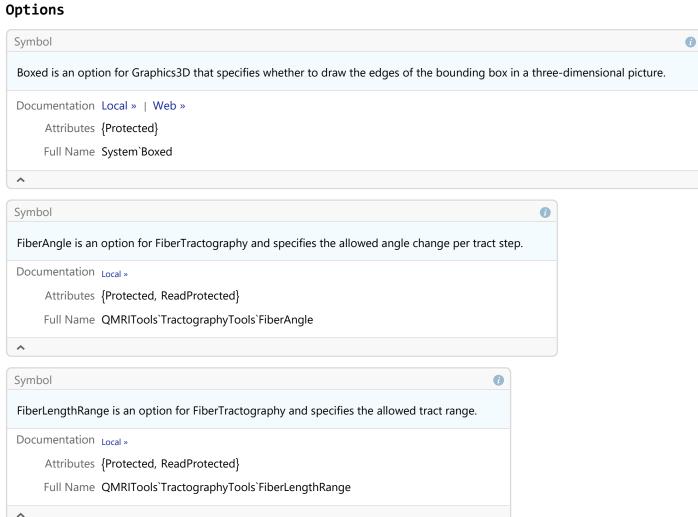


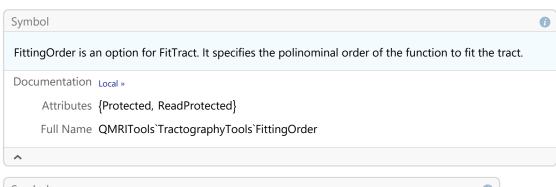


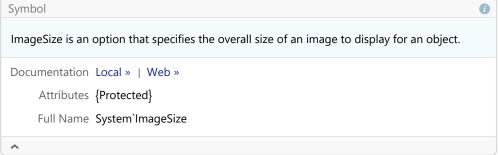




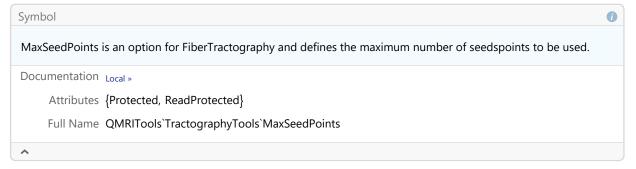


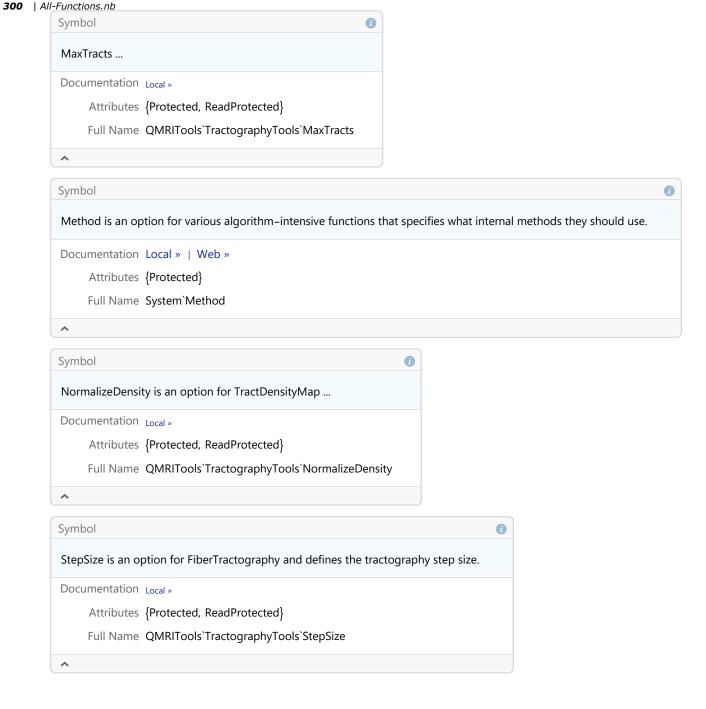


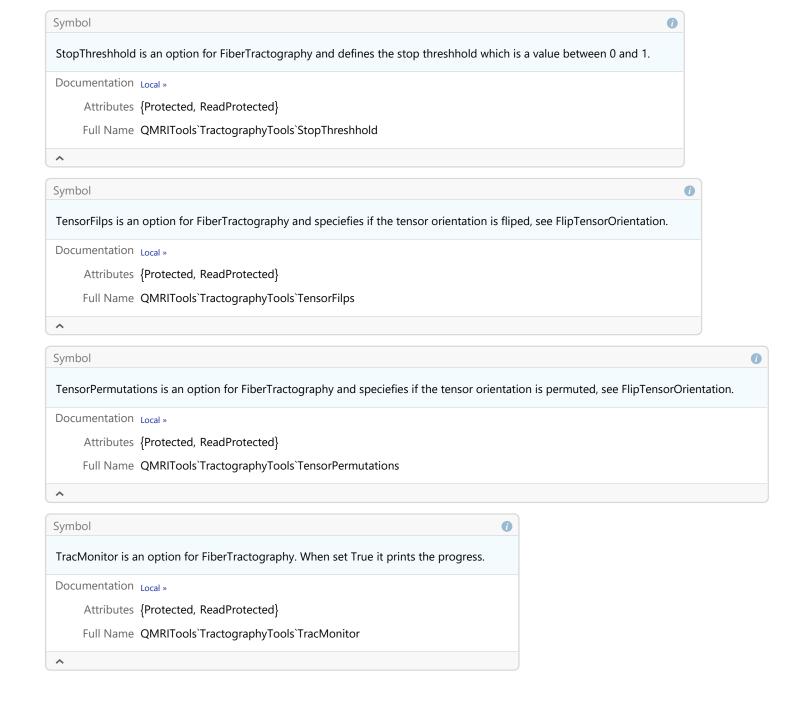


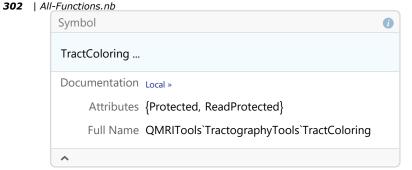








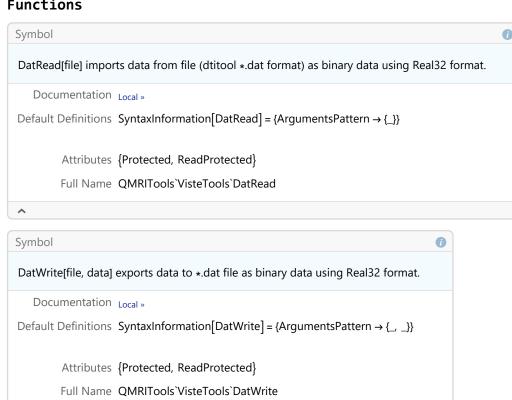




## **VisteTools**

### **Functions**

^



^

DTItoolExp[tensor, voxsize] exports tensor to {XX.dat, YY.dat, XZ.dat, XY.dat, XZ.dat, YZ.dat} and uses XX.dat as background and generates corresponding \*dti files.

DTItoolExp[tensor, voxsize, folder] exports tensor to {XX.dat, YY.dat, XZ.dat, XY.dat, XZ.dat} to the given folder and uses XX.dat as background and generates corresponding \*dti files.

DTItoolExp[tensor, voxsize, folder, add] exports tensor to {XX.dat, YY.dat, ZZ.dat, XY.dat, XZ.dat,

YZ.dat} to the given folder and uses XX.dat as background and generates corresponding \*dti files adds - add to the filenames.

DTItoolExp[back, tensor, voxsize] exports background to back.dat and tensor to {XX.dat, YY.dat, XZ.dat, XY.dat, XZ.dat} and generates corresponding \*dti files.

DTItoolExp[back, tensor, voxsize, folder] exports background to back.dat and tensor to {XX.dat, YY.dat, XZ.dat, XY.dat, XZ.dat} to the given folder and generates corresponding \*dti files.

DTItoolExp[back, tensor, voxsize, folder, add] exports background to back.dat and tensor to

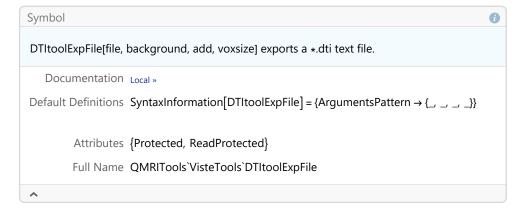
{XX.dat, YY.dat, ZZ.dat, XY.dat, XZ.dat, YZ.dat} to the given folder and generates corresponding \*dti files and adds – add to the filenames.

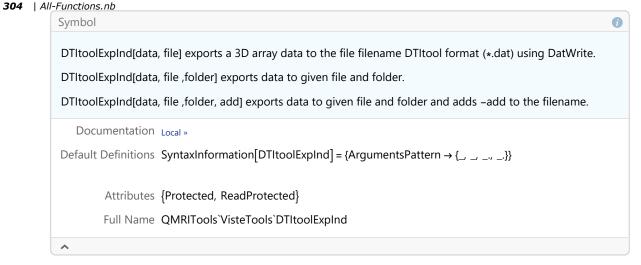
Documentation Local »

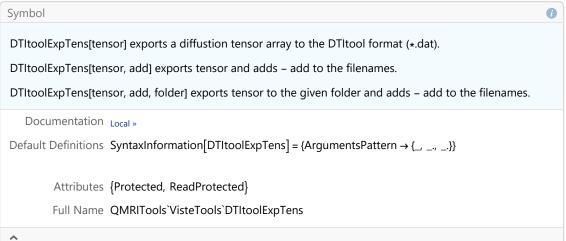
Default Definitions SyntaxInformation[DTItoolExp] = {ArgumentsPattern → {\_ , \_ , \_ , \_ , \_ , \_ }}

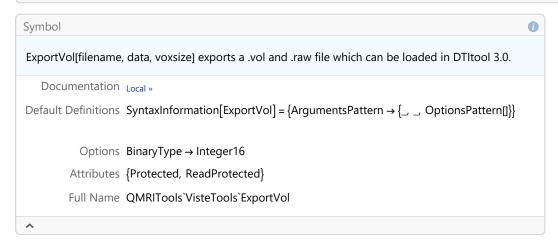
Attributes {Protected, ReadProtected}

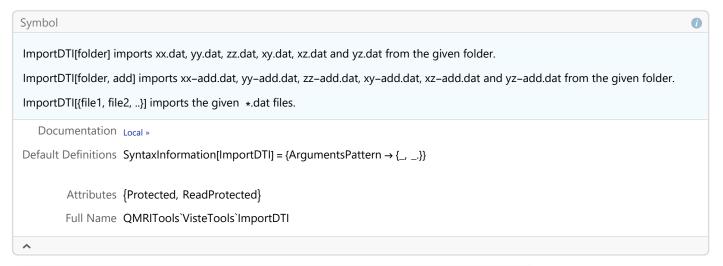
Full Name QMRITools`VisteTools`DTItoolExp

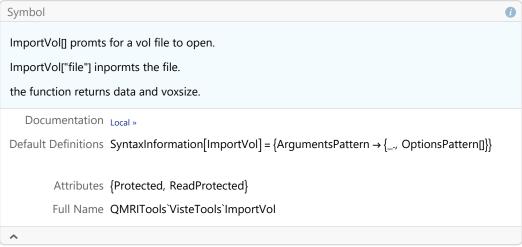


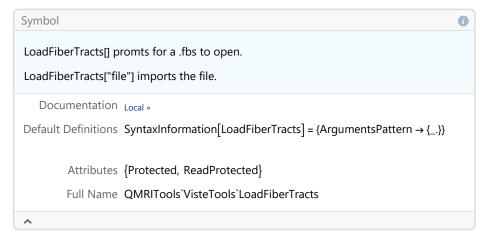












## **Options**

