PDECO - Input Options

A summary of all possible input options for PDECO. The options are called 'opts'. This has three substructures 'optsPhys','optsNum' and 'optsPlot'. These are detailed in the following.

1 optsPhys

There are three sub-structures in this structure. They are 'ProbSpecs', 'DataIn' and 'Params'.

1.1 optsPhys.optsPhysFW

The structure 'optsPhys.optsPhysFW' contains the following substructures: 'ProbSpecs', 'DataInput' and 'Params'.

1.1.1 ProbSpecs

The structure 'ProbSpecs' contains all problem specifications that are needed to specify a given forward problem to solve. The input options are summarized in the table below.

Input Name	'BCFunStr'
Description	Determines boundary conditions
Options	'ComputeDirichletBCs', 'ComputeNeumannBCs', 'ComputeMixedBCs' takes inputs: 'rho', 'rhoflux', 'bound', 'normal', 'this'. parameter 'eps' for Dirichlet Contribution in MixedBCs (See 'Params')

Input Name	'PDERHSStr'
Description	Determines which PDE is solved, all can include interaction term, switched on with 'gamma' (see 'Params')
Options 1D	'D_Force': Diffusion, Force Control AD_Force: Advection Diffusion, Force Control (inc. Vext, Force) AD_Forcefl: Advection Diffusion, Force Control (inc. Vext, Force, wFlow) AD_Flow: Advection Diffusion, Flow Control (only Flow term) AD_Flowf: Advection Diffusion, Flow Control (Flow term and Force) AD_FlowfVext: Advection Diffusion, Flow Control (Flow term, Force, Vext)
Options 2D	These remain the same for 1D and 2D
Note:	To include a new PDE see separate table below.

Input Name	'ComputeNorm'
Description	Specifies in which norm errors are computed within the solver. Takes function name as input. Needs to have syntax ComputeNorm(fNew,fOld,SInt,TInt). (SInt: Space Integration TInt: Time Integration)
Options	'ComputeRelL2LinfNorm': Relative Norm, L2 space, Linf time. 'ComputeRelPWNorm': Relative Norm, Pointwise errors. 'ComputeL1Norm': Absolute Norm, L1 space and time.

Naming conventions for adding PDEs

Two things to be aware of:
- if your gradient equation is different than force/flow control, you need to update this in the code
- generally, your PDERHSFlag needs to be added inside the code too

The naming convention for adding new PDEs into the code is:
- for a forward problem (also the forward problem during the optimization – it'll be the same both times)
'ComputeFW' + string to specify the problem. - for the adjoint equation
'ComputeOptP' + the SAME string that specifies the FW problem
slightly differs in 2D – see table below

	Adding new PDEs into PDECO:
1D FW (& rho Opt) Input arguments:	Naming convention of the function: 'ComputeFW'&'PDERHSStr': this, RHSInput
1D Adjoint Equation Input arguments:	Naming convention of the function: 'ComputeOptp'& 'PDERHSStr'. this, pRHSInput
2D FW (& rho Opt) Input arguments:	Naming convention of the function: 'ComputeFW2D'&'PDERHSStr' this, RHSInput
2D Adjoint Equation Input arguments:	Naming convention of the function:'ComputeOptp2D'& 'PDERHSStr' this, pRHSInput

1.1.2 optsPhysFW.DataInput

This structure includes all the input data to the solver. It can take the input as matrices or as function name. For each variable, the code checks whether the variable name exists as input. If it does, this will be used in the code, and interpolated where necessary. If it does not exist, then the field is created and filled by the given function. If all variables are specified by the function, then the only input in 'DataIn' is the function name! The below table illustrates the

options.

Input Name	Input Description	Options
testFun	Function Name of input function	e.g. 'AD_Flow_Neumann_Exact'. Needs all the below variables in output stru Set variable to zero if not needed.
DataRecompConv	Decides whether to check for datastorage results for the convolution matrix	true/ false
DataRecompFW	Decides whether to check for datastorage results for the forward problem	true/ false
Optional:		
rhoIC	Initial Condition for rho	Input vector $(1 \times N)$ or not existent.
pIC	Final Time Condition for p	Input vector $(1 \times N)$ or not existent.
wFlow	Flow Control term	Input Matrix $(n \times N)$ or not existent.
wForce	Force Control term	Input Matrix $(n \times N)$ or not existent.
Force	Additional Force term	Input Matrix $(n \times N)$ or not existent.
Flow	Additional Flow term	Input Matrix $(n \times N)$ or not existent.
Vext	External Potential term	Input Matrix $(n \times N)$ or not existent.

Note that the 'DataRecomp...' flags are there to determine whether to recompute your function (true) or whether to find a result in the datastorage folder (false). If nothing is specified, the default is 'false'.

1.1.3 optsPhysFW.Params

Here, any relevant parameters for the FW problem are specified.

Input Name	Description
beta	Regularization parameter in optimization
gamma	Magnitude of particle interaction term.
D0	Diffusion Coefficient.
eps	Contribution of Dirichlet term to Mixed BCs.
other	Additional input structure to 'testFun'.

1.2 optsPhys.optsPhysOpt

In this structure we have the inputs that are only necessary for the optimization but not for the forward solution. The optimization problem still gets all the inputs for the forward problem. The structures are: 'OptSolver', and 'OptDataInput'.

1.2.1 OptSolver

All things that are needed to specify the optimization solver are specified here.

Input Name	'SolverFlag'
Description	Choosing the solver for Optimization
Options	'fsolve': Inbuilt MatLab solver and with Multiple Shooting. 'Picard': Picard update and Multiple Shooting 'FixPt': Picard update/ Fixed Point iteration, no shooting

Input Name	'AdaSolverStr'
Description	Option to make Picard or Fixed Point Algorithm adaptive.
Options	Input is a function name, Function input 'Err' and 'lambda'. 'Adaptive': will change 'lambda' (see 'Params') to be adaptive '[ ]', or exclusion of this option will leave 'lambda' static.

Other input:

lambda	Mixing rate of old and new solution in 'Picard' and 'FixPt' solvers.
BCFunOptStr	In case the adjoint has different BCs from rho (such as in non-zero Dirichlet case)
OptsTols	Tolerances for the optimization solver (structure - see below).

These are the tolerances for the different optimization solvers.

Location of usage	Input Name	Description
fsolve	'OptsTols.FunTol'	Function Tolerance
fsolve	'OptsTols.OptiTol'	Optimality Tolerance
fsolve	'OptsTols.StepTol'	Stepsize Tolerance
Picard, FixPt	'OptsTols.ConsTol'	Consistency Tolerance Consistency condition for convergence.

OptDataInput

The DataRecompOpt option determines whether datastorage checks for an existing solution or whether it recomputes the solution.

Input Name	Description	Input Format
OptirhoIG	Initial guess for Optimization	Input Matrix ($n \times N$), not existent, or 'rhoFW' to call forward result as IG
OptipIG	Initial guess for Optimization	Input Matrix ($n \times N$) or not existent.
DataRecompOpt	whether to recompute data	true/false

1.3 optsPhys.V2Num/optsNum.V2Num

'V2Num' is part of both 'optsPhys' and 'optsNum'. It contains the Kernel function for the particle interaction term and takes additional parameters.

Input Name	Description
'V2'	Takes a function name, e.g. 'ComputeGaussian'. This function takes the inputs 'V2Num' and 'y', the points.
'alpha'	A parameter for the function.

2 optsNum

This structure has four (five) substructures: 'PhysArea', 'PlotArea'. 'TimeArea' and 'Tols' (and 'V2Num' - see above).

2.1 optsNum.PhysArea

Specifies the physical spacial domain on which computations should be carried out.

Dimension	Shape	Number of space points	Boundary
1D	'SpectralLine'	'N'	'yMin', 'yMax'
2D	'Box'	'N' (input as e.g. $[50, 50]$)	'y1Min', 'y1Max', 'y2Min', 'y2Max'

2.2 optsNum.PlotArea

Specifies plotting points and area.

Dimension	Number of plotting points	Boundary
1D	'N'	'yMin', 'yMax'
2D	'N1', 'N2'	'y1Min', 'y1Max', 'y2Min', 'y2Max'

2.3 optsNum.TimeArea

Specifies the time domain on which computations should be carried out.

Boundary left	Boundary right	Number of time points
't0'	'TMax'	'n'

2.4 optsNum.Tols

Solver tolerances etc.

Location of usage	Input Name	Description
ODE solver	'AbsTol'	Absolute Tolerance
ODE solver	'RelTol'	Relative Tolerance

3 optsPlot

This structure is independent of the forward or optimization computation. It contains the following:

'plotTimes'	plotting times the solution is interpolated on
'VarStruct'	structure that specifies the variables to be plotted and the location on 'subplot'. up to 8 locations specifiable.
'VarStruct' Inputs: Strings	A string followed by a number to specify the location 'rhoIC', 'pIC', 'OptirhoIG', 'OptipIG', 'rhoFW', 'rhoOpt', 'pOpt' 'wFW', 'wOpt', 'rhoHat', 'Force', Flow', 'Vext'

There is in principle also an option to supply a matrix to be plotted. Then you can't specify the location on the plot but instead give a structure 'VecOpts' containing 'VecOpts.titleName' (title of plot), 'Vecopts.yLabel' (yLabel of plot).
Example:
VarStr = struct('OptirhoIG',1,'OptipIG',2,M,VecOpts);
where M is the matrix you are interested in plotting.