

Benchmark solvers for solving whole body metabolic models

Authors: Ronan M.T. Fleming, University of Galway

Reviewers:

Introduction

Compare the time taken to solve different formulations of constraint-based modelling problems involving whole body metabolic models with different solvers and different methods for each solver with the option to repeat the analysis to compute mean and variance of solution times.

EQUIPMENT SETUP

Initialize the COBRA Toolbox.

Please ensure that The COBRA Toolbox has been properly installed, and initialized using the initCobraToolbox function.

```
if 0 %set to true if your toolbox has not been initialised
    initCobraToolbox(false) % false, as we don't want to update
end
```

PROCEDURE

Define the location to save your results

```
if 1
    resultsFolder = '~/drive/sbgCloud/projects/variationalKinetics/
results/WBM/';
else
    resultsFolder = pwd;
end
```

Load whole body metabolic model - change this to suit your own setup.

```
modelToUse = 'Harvey';
%modelToUse = 'Harvetta';
driver_loadBenchmarkWBMsolvers
```

Set parameters for benchmark

Model perturbation parameters

```
%model.ub(model.c~=0)=inf;
clear param T T0
param.replaceLargeBoundsWithInf=1;
```

```
param.relaxTightBounds=1;
param.relaxTightBounds_lowerExponent = 3; %the minimum difference between
ub_j and lb_j is 10^(param.relaxTightBounds_lowerExponent)
param.relaxTightBounds_higherExponent = 10;
param.setUpperBoundOnObjectiveToInf=1;
```

COBRA toolbox & solver parameters

```
param.printLevel = 0; % {(0),1,2} 1 output from optimiseVKmode, 2 also
output from solver
param.feasTol = 1e-5;
param.optTol = 1e-5;
param.lifted = 1;
param.multiscale = 1;
param.debug = 0;
```

Set the maximum time limit allowed to solve a single instance. Useful for eliminating slow instances in a large batch of trials.

```
param.timelimit = 200;
```

Select whether to compare one or a set of solvers

```
compareSolvers = 1;
```

Select whether to compare one or a set of different formulations of constraint-based modelling problems involving whole body metabolic models.

```
compareSolveWBMmethods = 1;
```

Select whether to compare one or a set of available methods (algorithms) for each solver

```
compareSolverMethods = 1;
```

Define the number of times to replicate the same formulation, solver, method combination.

```
nReplicates = 2;
```

Display and (optionally) modify properties of the whole body model that may effect solve time

```
[nMet,nRxn]=size(model.S)
```

```
nMet =
58095
nRxn =
83395
```

Identify large bounds not at the maximum

```
boundMagnitudes = [abs(model.lb);abs(model.ub)];
boundMagnitudes(~isfinite(boundMagnitudes))=0;
largestMagnitudeBound = max(boundMagnitudes);
```

```

fprintf('%.g%s\n',
(nnz(largestMagnitudeBound==[abs(model.lb);abs(model.ub)])*100)/
(length(model.lb)*2), ' = percent of bounds at maximum')

```

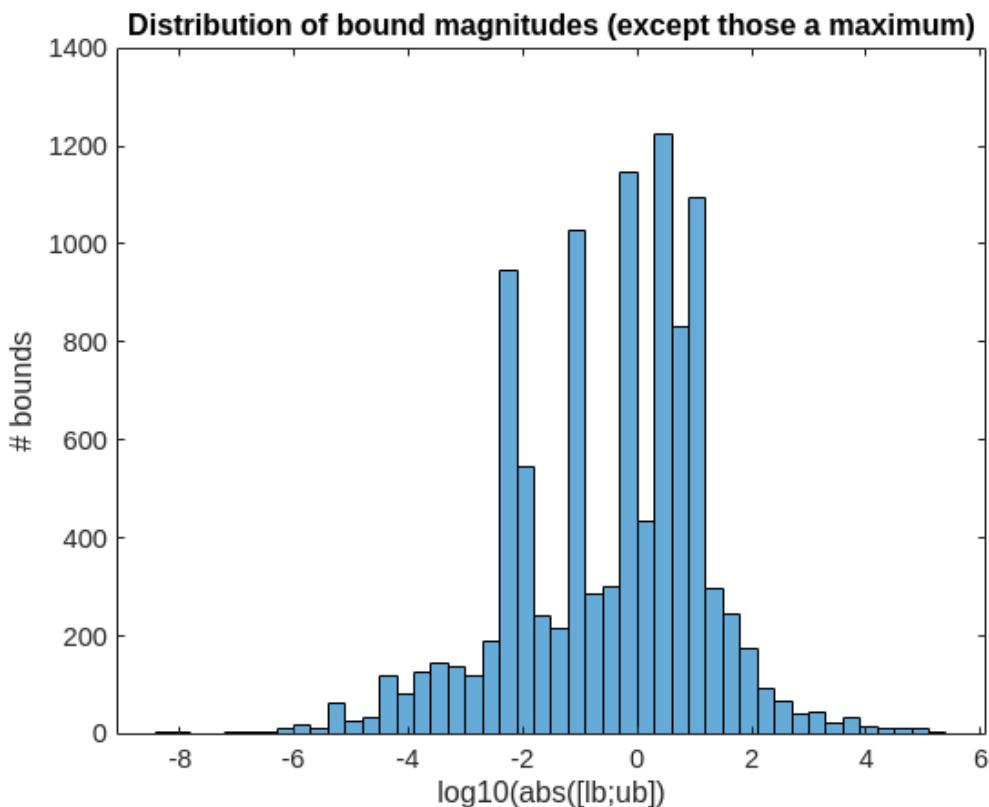
62.2112 = percent of bounds at maximum

Display bounds that are not at the maximum

```

if 1
    figure
    histogram(log10(boundMagnitudes(boundMagnitudes~=largestMagnitudeBound &
boundMagnitudes~=0)))
    xlabel('log10(abs([lb;ub]))')
    ylabel('# bounds')
    title('Distribution of bound magnitudes (except those a maximum)')
end

```



Replace large bounds with inf or -inf. This is a good idea. Better to leave this option on.

```

if param.replaceLargeBoundsWithInf && largestMagnitudeBound>1e3
    model.lb(-largestMagnitudeBound==model.lb)=-inf;
    model.ub(largestMagnitudeBound==model.ub)= inf;
end
boolMagnitudes = boundMagnitudes~=largestMagnitudeBound & boundMagnitudes~=0
& boundMagnitudes<1e-4;
boolRxns = boolMagnitudes(1:nRxn) | boolMagnitudes(nRxn+1:2*nRxn);

```

Optionally, print the bounds for reactions with small magnitude

```
if 0
    printFluxBounds(model,model.rxns(boolRxns))
end
fprintf('%g%s\n',nnz(boolRxns)*100/length(boolRxns), ' = percent of bounds
with magnutide less than 1e-4')
```

0.389712 = percent of bounds with magnutide less than 1e-4

Optionally, print the bounds for reactions with small difference

```
boundDifference = model.ub - model.lb;
bool = length(model.rxns);
Z = table(boundDifference,model.rxns,model.rxnNames,'VariableNames',
{'boundDifference','rxns','rxnNames'});
if any(boundDifference<0)
    error(['lb > ub for ' num2str(nnz(boundDifference)) ' reactions'])
end
boolDifference = boundDifference<1e-5 & boundDifference~=0;
Z = sortrows(Z(boolDifference,:),'boundDifference');
if 0
    printFluxBounds(model,Z.rxns,1)
end
fprintf('%g%s\n',nnz(boolRxns)*100/length(boolRxns), ' = percent of bounds
with difference (ub - lb) less than 1e-5')
```

0.389712 = percent of bounds with difference (ub - lb) less than 1e-5

```
forwardBoolDifference = boolDifference & model.lb>=0 & model.ub>0;
reverseBoolDifference = boolDifference & model.lb<0 & model.ub<=0;
reversibleBoolDifference = boolDifference & model.lb<0 & model.ub>0;

if any((forwardBoolDifference | reverseBoolDifference |
reversibleBoolDifference) ~= boolDifference)
    error('missing bool difference')
end
```

Optionally relax bounds that are very tight

```
if param.relaxTightBounds
    modelOld=model;
    done=false(nRxn,1);
    for
x=param.relaxTightBounds_higherExponent:-1:param.relaxTightBounds_lowerExpone
nt
        %calculate the difference between the bounds each time
        boundDifference = model.ub - model.lb;

        %forward
        bool = forwardBoolDifference & (boundDifference <= 10^(-x));
```

```

        model.ub(bool & ~done) = model.ub(bool & ~done)*(10^(x-
param.relaxTightBounds_lowerExponent+1));
        done = done | bool;

        %reverse
        bool = reverseBoolDifference & (boundDifference <= 10^(-x));
        model.lb(bool & ~done) = model.lb(bool & ~done)*(10^(x-
param.relaxTightBounds_lowerExponent+1));
        done = done | bool;

        %reversible
        bool = reversibleBoolDifference & (boundDifference <= 10^(-x));
        model.lb(bool & ~done) = model.lb(bool & ~done)*(10^((x-
param.relaxTightBounds_lowerExponent+1)/2));
        model.ub(bool & ~done) = model.ub(bool & ~done)*(10^((x-
param.relaxTightBounds_lowerExponent+1)/2));
        done = done | bool;

        %reset
        %done=false(nRxn,1);
    end
    fprintf('%g%s\n',nnz(done), [ ' = reactions with tight bounds relaxed to
at least' num2str(param.relaxTightBounds_lowerExponent) ' for ub - lb'])
    if 1
        printFluxBounds(model,Z.rxns,1)
    end
end

```

69 = reactions with tight bounds relaxed to at least3 for ub - lb			
Reaction ID	Lower Bound	Upper Bound	
BBB_ESTRADIOL[CSF]exp	0.000e+00	5.544e-03	Estradiol Glucuronide Transport via Bica
BBB_LEUKTRB4WCOOH[CSF]exp	0.000e+00	7.560e-03	Transport of W-Carboxy Leukotriene B4, A
BBB_LEUKTRB4WOH[CSF]exp	0.000e+00	7.560e-03	Transport of W-Hydroxyl Leukotriene B4,
EX_sphings[u]	0.000e+00	8.374e-03	Exchange of Sphingosine
EX_leuktrB4[u]	0.000e+00	1.231e-03	Exchange of Leukotriene B4
BBB_PRGNLONE[CSF]exp	0.000e+00	9.576e-03	Steroid SulfotransferaseTransport of Pre
BBB_LEUKTRC4[CSF]exp	0.000e+00	1.018e-03	Transport of Leukotriene C4 via Bicarbon
EX_prostgdf2[u]	0.000e+00	1.167e-03	Exchange of Prostaglandin F2Alpha
BBB_C14771[CSF]exp	0.000e+00	1.285e-03	Transport of 14, 15-EET, Active Transpor
EX_thyox_L[u]	0.000e+00	1.315e-03	Exchange of L-Thyroxine
EX_sphgn[u]	0.000e+00	1.337e-03	Exchange of Sphinganine
BBB_MEPI[CSF]exp	0.000e+00	2.419e-03	Metanephrine Secretion via Secretory Ves
EX_C05767[u]	0.000e+00	3.310e-03	Exchange of Uroporphyrin I
BBB_C04805[CSF]exp	0.000e+00	3.578e-03	Transport of 5(S)-HETE, Active Transport
EX_estriol[u]	0.000e+00	4.456e-03	Exchange of Estriol
Kidney_EX_no(e)_[bc]	-4.484e-03	0.000e+00	Exchange of Nitric Oxide (frombloodto[e])
EX_estradiol[u]	0.000e+00	4.922e-03	Exchange of Estradiol
BBB_BTN[CSF]exp	0.000e+00	5.902e-03	Biotinidase (Biotin)Biotinidase (Biotin)
EX_estrone[u]	0.000e+00	6.365e-03	Exchange of Estrone
BBB_CE2047[CSF]exp	0.000e+00	6.905e-03	Transport (ATP-Dependent) into Extracell
BBB_CE2049[CSF]exp	0.000e+00	7.560e-03	Transport (ATP-Dependent) into Extracell
BBB_12HARACHD[CSF]exp	0.000e+00	7.711e-03	Transport of 12 Hydroxy Arachidonic Acid
BBB_ANTH[CSF]exp	0.000e+00	7.762e-03	Transport of Anthranilate (BBB)
EX_pcholn204_hs[u]	0.000e+00	8.062e-03	"Exchange of 1-Eicosatetraenoylglyceroph
EX_C05298[u]	0.000e+00	8.699e-03	Exchange of 2-Hydroxyestrone
EX_C05301[u]	0.000e+00	9.335e-03	Exchange of 2-Hydroxyestradiol-17Beta

EX_tdchola[u]	0.000e+00	9.441e-03	Exchange of Taurochenodeoxycholate
BBB_PRGSTRN[CSF]exp	0.000e+00	1.008e-03	Progesterone Transport (BBB)
BBB_TSTSTERONE[CSF]exp	0.000e+00	1.008e-03	Glucuronidated Compound Transport
BBB_ANDRSTNDN[CSF]exp	0.000e+00	1.008e-03	Transport of Androst-4-Ene-3, 17-Dione,
BBB_DHEA[CSF]exp	0.000e+00	1.008e-03	Dehydroepiandrosterone Sulfate Transport
BBB_CE0955[CSF]exp	0.000e+00	1.124e-03	Transport of 6-Oxo-Prostaglandin F1Alpha
EX_C05302[u]	0.000e+00	1.231e-03	Exchange of 2-Methoxyestradiol-17Beta
EX_C05299[u]	0.000e+00	1.273e-03	Exchange of 2-Methoxyestrone
EX_dgchol[u]	0.000e+00	1.379e-03	Exchange of Chenodeoxyglycocholate
EX_argsuc[u]	0.000e+00	1.379e-03	Exchange of L-Arginosuccinic Acid
BBB_XOL27OH[CSF]exp	0.000e+00	1.411e-03	27 Hydroxy Cholesterol Transport
Diet_EX_adpcbl[d]	-4.332e-03	-2.888e-06	Diet_EX_adpcbl[d]
BBB_IM4AC[CSF]exp	0.000e+00	1.562e-03	Assumed Passive Diffusion into Extracellular
BBB_35CGMP[CSF]exp	0.000e+00	1.714e-03	35CGMP Nuclear Transport (BBB)
BBB_C14826[CSF]exp	0.000e+00	1.799e-03	Transport of 12 (13)-Epome, FATTPTranspor
BBB_XOL24OH[CSF]exp	0.000e+00	1.814e-03	Transport of (24S)-24-Hydroxycholesterol
EX_pcholhep_hs[u]	0.000e+00	1.846e-03	Exchange of 1-Heptadecanoylglycerophospho
EX_C05770[u]	0.000e+00	1.952e-03	Exchange of Coproporphyrin III
BBB_C14825[CSF]exp	0.000e+00	1.981e-03	Formation of 9 (10)-EpomeTransport of 9
EX_5htrp[u]	0.000e+00	1.986e-03	Exchange of 5-Hydroxy-L-Tryptophan
BBB_PROSTGF2[CSF]exp	0.000e+00	2.016e-03	Prostaglandin Uniport (BBB)
EX_adrn1[u]	0.000e+00	2.100e-03	Exchange of Adrenaline
EX_pcholste_hs[u]	0.000e+00	2.122e-03	Exchange of 1-Stearoylglycerophosphochol
BBB_3MOXTYR[CSF]exp	0.000e+00	2.280e-03	3-Methoxytyramine:Oxygen Oxidoreductase
BBB_LEUKTRB4[CSF]exp	0.000e+00	2.313e-03	Transport of Leukotriene B4Transport of
EX_fol[u]	0.000e+00	2.546e-03	Exchange of Folate
EX_aldstrn[u]	0.000e+00	2.970e-03	Exchange of Aldosterone
BBB_34DHPHA[CSF]exp	0.000e+00	3.024e-03	3, 4-Dihydroxyphenylacetate:Amet O-Methyl
EX_prostge2[u]	0.000e+00	4.031e-03	Exchange of Prostaglandin E2
EX_crtsl[u]	0.000e+00	4.456e-03	Exchange of Cortisol
BBB_34DHPHE[CSF]exp	0.000e+00	5.040e-03	3, 4-Dihydroxy-L-Phenylalanine Transport
BBB_5HTRP[CSF]exp	0.000e+00	5.040e-03	5-Hydroxy-L-Tryptophan Secretion via Sec
BBB_DOPA[CSF]exp	0.000e+00	5.040e-03	Dopamine Beta-MonooxygenaseL-Dopachrome
BBB_SRTN[CSF]exp	0.000e+00	5.040e-03	Acetyl Coenzyme A:Aralkylamine N-Acetyltyl
BBB_THMMP[CSF]exp	0.000e+00	5.040e-03	Thiamine Monophosphate Transport in via
BBB_NORMETE_L[CSF]exp	0.000e+00	5.040e-03	Export of normete_L[csf] from CSF to bloo
BBB_MHISTA[CSF]exp	0.000e+00	5.040e-03	Facilitated Diffusion Through Uniport Oo
BBB_CE2705[CSF]exp	0.000e+00	5.040e-03	Transport by Ent1/Ent2 into Extracellular
BBB_CE4890[CSF]exp	0.000e+00	5.040e-03	Facilitated Diffusion Through Uniport Oo
BBB_C09642[CSF]exp	0.000e+00	5.040e-03	Facilitated Diffusion Through Uniport Oo
Gall_EX_adrn1(e)_[bc]	-7.915e-03	-0.000e+00	Exchange of Adrenaline (frombloodto[e])
EX_34dhphe[u]	0.000e+00	8.487e-03	"Exchange of 3, 4-Dihydroxy-L-Phenylalan
Kidney_EX_leuktrD4(e)_[bc]	-9.342e-03	0.000e+00	Exchange of Leukotriene D4 (frombloodto[

```
disp(min(boundDifference(boundDifference~=0)))
```

1.0080e-05

Optionally, remove fixed upper bound on biomass reaction

```
% Optionally, relax bounds that are fixed for the objective
if param.setUpperBoundOnObjectiveToInf
    if any(contains(modelToUse,['Harvey','Harvetica'])) && nnz(model.c)==1
        biomassRxnAbbr = model.rxns{model.c~=0};
        if
            model.ub(ismember(model.rxns,biomassRxnAbbr))==model.ub(ismember(model.rxns,biomassRxnAbbr))
                if strcmp(model.osenseStr,'max')
                    model.ub(ismember(model.rxns,biomassRxnAbbr))=inf;
```

```

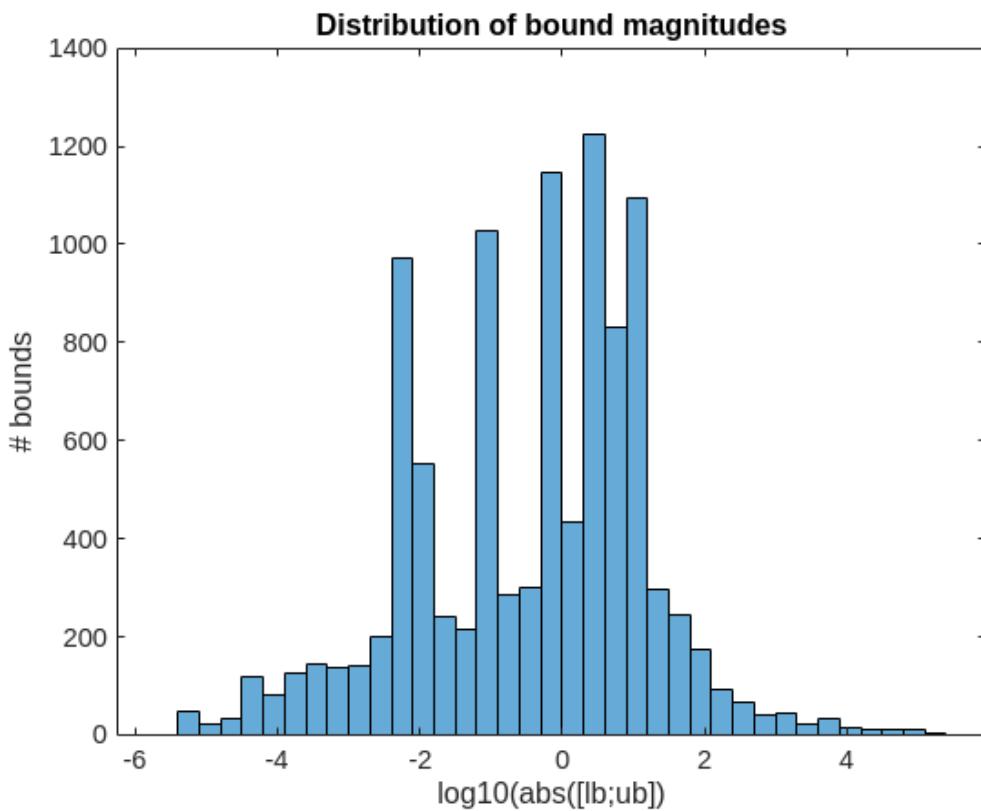
        else
            model.lb(ismember(model.rxn, biomassRxnAbbr))=-inf;
        end
    end
end

```

```

if 1
    boundMagnitudes = [abs(model.lb);abs(model.ub)];
    figure
    histogram(log10(boundMagnitudes))
    xlabel('log10(abs([lb;ub]))')
    ylabel('# bounds')
    title('Distribution of bound magnitudes')
end

```



Prepare a benchmark table, choose the solver and solve

```

VariableNames={'interface','solver','method','problem','model','stat','origSt
at','time','obj','f1','f2','f0'};
% Define the corresponding variable types
VariableTypes = {'string', 'string',
'string','string','string','double','string','double','double','double','doub
le','double'};

```

```
T = table('Size', [0
length(VariableNames)], 'VariableNames', VariableNames, 'VariableTypes',
VariableTypes);
```

Select the solvers to compare

```
if compareSolvers
    solvers = {'mosek','ibm_cplex','gurobi'};
    %solvers = { 'ibm_cplex','mosek','gurobi'} ;
    %solvers = { 'mosek','ibm_cplex','gurobi'} ;
else
    % Choose the solver
    % solvers = {'gurobi'} ;
    %solvers = {'ibm_cplex'} ;
    solvers = {'mosek'} ;
end
```

Select the formulations to compare

```
if compareSolveWBMmethods
    %solveWBMmethods = {'LP','QP','QRLP','QRQP','zero','oneInternal'} ;
    % solveWBMmethods = {'LP','QP'};%,'zero','oneInternal'} ;
    %solveWBMmethods = {'LP','oneInternal'} ;
    solveWBMmethods = {'QP','LP'} ;
else
    % Choose type of problem to solve
    solveWBMmethods = {'LP'} ;
    %solveWBMmethods = {'QP'} ;
    %solveWBMmethods = {'QRLP'} ;
    %solveWBMmethods = {'QRQP'} ;
end
```

Define the methods (algorithms) available for different solvers

```
if compareSolverMethods
    % CPLEX
    % 0      CPX_ALG_AUTOMATIC      Automatic: let CPLEX choose; default
    % 1      CPX_ALG_PRIMAL        Primal simplex
    % 2      CPX_ALG_DUAL          Dual simplex
    % 3      CPX_ALG_NET           Network simplex
    % 4      CPX_ALG_BARRIER        Barrier
    % 5      CPX_ALG_SIFTING       Sifting
    % 6      CPX_ALG_CONCURRENT    Concurrent (Dual, Barrier, and Primal in
opportunistic parallel mode; Dual and Barrier in deterministic parallel mode)
    %https://www.ibm.com/docs/en/icos/12.10.0?topic=parameters-algorithm-continuous-linear-problems
    cplexLPMETHODS
= {'AUTOMATIC','PRIMAL','DUAL','NETWORK','BARRIER','SIFTING','CONCURRENT'} ;
    % 0      CPX_ALG_AUTOMATIC      Automatic: let CPLEX choose; default
```

```

% 1      CPX_ALG_PRIMAL      Use the primal simplex optimizer.
% 2      CPX_ALG_DUAL        Use the dual simplex optimizer.
% 3      CPX_ALG_NET         Use the network optimizer.
% 4      CPX_ALG_BARRIER      Use the barrier optimizer.
% 6      CPX_ALG_CONCURRENT   Use the concurrent optimizer.
% https://www.ibm.com/docs/en/icos/12.10.0?topic=parameters-algorithm-
continuous-quadratic-optimization
cplexQPMMethods
={ 'AUTOMATIC' , 'PRIMAL' , 'DUAL' , 'NETWORK' , 'BARRIER' , 'CONCURRENT' } ;

% Mosek

% MSK_IPAR_OPTIMIZER
%     The parameter controls which optimizer is used to optimize the task.
%     Default "FREE"
%     Accepted "FREE", "INTPNT", "CONIC", "PRIMAL_SIMPLEX",
"DUAL_SIMPLEX", "FREE_SIMPLEX", "MIXED_INT"
%     Example param.MSK_IPAR_OPTIMIZER = 'MSK_OPTIMIZER_FREE'
mosekLPMMethods ={'FREE', 'INTPNT', 'CONIC', 'PRIMAL_SIMPLEX',
'DUAL_SIMPLEX', 'FREE_SIMPLEX'} ;
mosekQPMMethods ={'FREE', 'INTPNT'} ; %CONIC not yet encoded in
solveCobraQP

% Gurobi
% https://www.gurobi.com/documentation/current/refman/method.html
% Algorithm used to solve continuous models
% Algorithm used to solve continuous models or the initial root
relaxation of a MIP model. Options are:
gurobiLPMMethods =
{ 'AUTOMATIC' , 'PRIMAL' , 'DUAL' , 'BARRIER' , 'CONCURRENT' , 'DETERMINISTIC_CONCURRENT'
} ;
gurobiQPMMethods = { 'AUTOMATIC' , 'PRIMAL' , 'DUAL' , 'BARRIER' } ;
else
if 0
    gurobiLPMMethods={ 'BARRIER' } ;
    gurobiQPMMethods={ 'BARRIER' } ;
    cplexLPMMethods={ 'BARRIER' } ;
    cplexQPMMethods={ 'BARRIER' } ;
    mosekLPMMethods = { 'CONIC' } ;
    mosekQPMMethods = { 'FREE' } ;
else
    gurobiLPMMethods={ 'AUTOMATIC' } ;
    gurobiQPMMethods={ 'AUTOMATIC' } ;
    cplexLPMMethods={ 'AUTOMATIC' } ;
    cplexQPMMethods={ 'AUTOMATIC' } ;
    mosekLPMMethods = { 'FREE' } ;
    mosekQPMMethods = { 'FREE' } ;
end
end

```

Set the min norm weight for QP problems

```
minNormWeight = 1e-4;  
%model.c(:)=0;
```

Solve the ensemble of instances

```
for ind = 1:nReplicates  
    for i = 1:length(solveWBMmethods)  
        param.solveWBMmethod = solveWBMmethods{i};  
        switch param.solveWBMmethod  
            case 'LP'  
                param.minNorm = [];  
            case 'QP'  
                param.minNorm = minNormWeight;  
            case 'QRLP'  
                param.minNorm = [];  
            case 'QRQP'  
                param.minNorm = minNormWeight;  
            case 'zero'  
                param.minNorm = 'zero';  
            case 'oneInternal'  
                if isfield(model, 'SConsistentRxnBool')  
                    param.minNorm = 'oneInternal';  
                else  
                    error('param.solveWBMmethod = oneInternal cannot be  
implemented as model.SConsistentRxnBool is missing')  
                end  
            end  
  
        for j = 1:length(solvers)  
  
            param.solver=solvers{j};  
  
            switch param.solver  
                case 'gurobi'  
                    if any(strcmp(param.solveWBMmethod,  
{'LP','zero','oneInternal'}))  
                        solverMethods = gurobiLPMETHODS;  
                    else  
                        solverMethods = gurobiQPMETHODS;  
                    end  
                case 'ibm_cplex'  
                    if any(strcmp(param.solveWBMmethod,  
{'LP','zero','oneInternal'}))  
                        solverMethods = cplexLPMETHODS;  
                    else  
                        solverMethods = cplexQPMETHODS;  
                    end  
                case 'mosek'
```

```

        if any(strcmp(param.solveWBMmethod,
{'LP','zero','oneInternal'}))
            solverMethods = mosekLPMMethods;
        else
            solverMethods = mosekQPMethods;
        end
    end

    % Solve a problem with selected solver and each method available
    % to that solver
    for k=1:length(solverMethods)
        if any(strcmp(param.solveWBMmethod,
{'LP','zero','oneInternal'}))
            param.lpmethod = solverMethods{k};
            if isfield(param,'qpmethod')
                param = rmfield(param,'qpmethod');
            end
        else
            param.qpmethod = solverMethods{k};
            if isfield(param,'lpmethod')
                param = rmfield(param,'lpmethod');
            end
        end
        tic
        try
            %return
            solution = optimizeCbModel(model,'min',
param.minNorm,1,param);
            catch ME
                disp('-----fail-----')
                disp('Error Message:')
                disp(ME.message)
                disp(param.solver)
                disp(solverMethods{k})
                disp(param.solveWBMmethod)
                disp(param)
                disp('-----fail-----')
            end
            T = [T; {'optimizeCbModel', param.solver, solverMethods{k},
param.solveWBMmethod, modelToUse, solution.stat,{solution.origStat},toc,
{solution.obj},{solution.f1},{solution.f2},{solution.f0}}];
            %display(T(end,:));
        end
        display(T)
    end
end

```

1x12 table

interface	solver	method	problem	model	stat	origStat	time
-----------	--------	--------	---------	-------	------	----------	------

"optimizeCbModel"	"mosek"	"FREE"	"QP"	"Harvey"	1	"OPTIMAL"	62.06
1x12 table							
interface	solver	method	problem	model	stat	origStat	time
"optimizeCbModel"	"mosek"	"INTPNT"	"QP"	"Harvey"	1	"OPTIMAL"	62.152
T =	2x12 table						...
	interface	solver	method	problem	model	stat	origStat
1	"optimizeCbModel"	"mosek"	"FREE"	"QP"	"Harvey"	1	"OPTIMAL"
2	"optimizeCbModel"	"mosek"	"INTPNT"	"QP"	"Harvey"	1	"OPTIMAL"
1x12 table							
interface	solver	method	problem	model	stat	origStat	time
"optimizeCbModel"	"ibm_cplex"	"AUTOMATIC"	"QP"	"Harvey"	3	"non-optimal"	30.
1x12 table							
interface	solver	method	problem	model	stat	origStat	
"optimizeCbModel"	"ibm_cplex"	"PRIMAL"	"QP"	"Harvey"	-1	"time limit exceeded"	
1x12 table							
interface	solver	method	problem	model	stat	origStat	
"optimizeCbModel"	"ibm_cplex"	"DUAL"	"QP"	"Harvey"	-1	"time limit exceeded"	
1x12 table							
interface	solver	method	problem	model	stat	origStat	
"optimizeCbModel"	"ibm_cplex"	"NETWORK"	"QP"	"Harvey"	-1	"time limit exceeded"	
1x12 table							
interface	solver	method	problem	model	stat	origStat	time
"optimizeCbModel"	"ibm_cplex"	"BARRIER"	"QP"	"Harvey"	3	"non-optimal"	30.37
1x12 table							
interface	solver	method	problem	model	stat	origStat	
"optimizeCbModel"	"ibm_cplex"	"CONCURRENT"	"QP"	"Harvey"	-1	"time limit exceeded"	
T =	8x12 table						...

	interface	solver	method	problem	model	stat	origStat
1	"optimizeCbModel"	"mosek"	"FREE"	"QP"	"Harvey"	1	"OPTIMAL"
2	"optimizeCbModel"	"mosek"	"INTPNT"	"QP"	"Harvey"	1	"OPTIMAL"
3	"optimizeCbModel"	"ibm_cplex"	"AUTOMATIC"	"QP"	"Harvey"	3	"non-optimal"
4	"optimizeCbModel"	"ibm_cplex"	"PRIMAL"	"QP"	"Harvey"	-1	"time limit exceeded"
5	"optimizeCbModel"	"ibm_cplex"	"DUAL"	"QP"	"Harvey"	-1	"time limit exceeded"
6	"optimizeCbModel"	"ibm_cplex"	"NETWORK"	"QP"	"Harvey"	-1	"time limit exceeded"
7	"optimizeCbModel"	"ibm_cplex"	"BARRIER"	"QP"	"Harvey"	3	"non-optimal"
8	"optimizeCbModel"	"ibm_cplex"	"CONCURRENT"	"QP"	"Harvey"	-1	"time limit exceeded"

1×12 table

interface	solver	method	problem	model	stat	origStat	time
"optimizeCbModel"	"gurobi"	"AUTOMATIC"	"QP"	"Harvey"	-1	"TIME_LIMIT"	201.78

1×12 table

interface	solver	method	problem	model	stat	origStat	time
"optimizeCbModel"	"gurobi"	"PRIMAL"	"QP"	"Harvey"	-1	"TIME_LIMIT"	200.18

1×12 table

interface	solver	method	problem	model	stat	origStat	time
"optimizeCbModel"	"gurobi"	"DUAL"	"QP"	"Harvey"	-1	"TIME_LIMIT"	200.18

1×12 table

interface	solver	method	problem	model	stat	origStat	time
"optimizeCbModel"	"gurobi"	"BARRIER"	"QP"	"Harvey"	-1	"TIME_LIMIT"	200.2

T = 12×12 table

	interface	solver	method	problem	model	stat	origStat
1	"optimizeCbModel"	"mosek"	"FREE"	"QP"	"Harvey"	1	"OPTIMAL"
2	"optimizeCbModel"	"mosek"	"INTPNT"	"QP"	"Harvey"	1	"OPTIMAL"
3	"optimizeCbModel"	"ibm_cplex"	"AUTOMATIC"	"QP"	"Harvey"	3	"non-optimal"
4	"optimizeCbModel"	"ibm_cplex"	"PRIMAL"	"QP"	"Harvey"	-1	"time limit exceeded"
5	"optimizeCbModel"	"ibm_cplex"	"DUAL"	"QP"	"Harvey"	-1	"time limit exceeded"
6	"optimizeCbModel"	"ibm_cplex"	"NETWORK"	"QP"	"Harvey"	-1	"time limit exceeded"
7	"optimizeCbModel"	"ibm_cplex"	"BARRIER"	"QP"	"Harvey"	3	"non-optimal"

	interface	solver	method	problem	model	stat	origStat
8	"optimizeCbModel"	"ibm_cplex"	"CONCURRENT"	"QP"	"Harvey"	-1	"time limit exceeded"
9	"optimizeCbModel"	"gurobi"	"AUTOMATIC"	"QP"	"Harvey"	-1	"TIME_LIMIT"
10	"optimizeCbModel"	"gurobi"	"PRIMAL"	"QP"	"Harvey"	-1	"TIME_LIMIT"
11	"optimizeCbModel"	"gurobi"	"DUAL"	"QP"	"Harvey"	-1	"TIME_LIMIT"
12	"optimizeCbModel"	"gurobi"	"BARRIER"	"QP"	"Harvey"	-1	"TIME_LIMIT"

1x12 table

interface	solver	method	problem	model	stat	origStat	time	obj
"optimizeCbModel"	"mosek"	"FREE"	"LP"	"Harvey"	1	"OPTIMAL"	23.359	{ [2.65]

1x12 table

interface	solver	method	problem	model	stat	origStat	time	obj
"optimizeCbModel"	"mosek"	"INTPNT"	"LP"	"Harvey"	1	"OPTIMAL"	23.298	{ [2.]

OPTIMAL

1x12 table

interface	solver	method	problem	model	stat	origStat	time	obj
"optimizeCbModel"	"mosek"	"CONIC"	"LP"	"Harvey"	1	"OPTIMAL"	16.838	{ [2.7]

Mosek returned an error or warning, open the following link in your browser:

https://docs.mosek.com/latest/toolbox/response-codes.html#mosek.rescode.trm_max_time

1x12 table

interface	solver	method	problem	model	stat	origStat
"optimizeCbModel"	"mosek"	"PRIMAL_SIMPLEX"	"LP"	"Harvey"	-1	"DUAL_FEASIBLE & MSK_RES_TRM_MAX_TIME"

Mosek returned an error or warning, open the following link in your browser:

https://docs.mosek.com/latest/toolbox/response-codes.html#mosek.rescode.trm_max_time

1x12 table

interface	solver	method	problem	model	stat	origStat
"optimizeCbModel"	"mosek"	"DUAL_SIMPLEX"	"LP"	"Harvey"	-1	"UNKNOWN & MSK_RES_TRM_MAX_TIME"

Mosek returned an error or warning, open the following link in your browser:

https://docs.mosek.com/latest/toolbox/response-codes.html#mosek.rescode.trm_max_time

1x12 table

interface	solver	method	problem	model	stat	origStat
"optimizeCbModel"	"mosek"	"FREE_SIMPLEX"	"LP"	"Harvey"	-1	"UNKNOWN & MSK_RES_TRM_MAX_TIME"

T = 18x12 table

	interface	solver	method	problem	model	stat	origStat
1	"optimizeCbModel"	"mosek"	"FREE"	"QP"	"Harvey"	1	"OPTIMAL"

	interface	solver	method	problem	model	stat	origStat
2	"optimizeCbModel"	"mosek"	"INTPNT"	"QP"	"Harvey"	1	"OPTIMAL"
3	"optimizeCbModel"	"ibm_cplex"	"AUTOMATIC"	"QP"	"Harvey"	3	"non-optimal"
4	"optimizeCbModel"	"ibm_cplex"	"PRIMAL"	"QP"	"Harvey"	-1	"time limit exceeded"
5	"optimizeCbModel"	"ibm_cplex"	"DUAL"	"QP"	"Harvey"	-1	"time limit exceeded"
6	"optimizeCbModel"	"ibm_cplex"	"NETWORK"	"QP"	"Harvey"	-1	"time limit exceeded"
7	"optimizeCbModel"	"ibm_cplex"	"BARRIER"	"QP"	"Harvey"	3	"non-optimal"
8	"optimizeCbModel"	"ibm_cplex"	"CONCURRENT"	"QP"	"Harvey"	-1	"time limit exceeded"
9	"optimizeCbModel"	"gurobi"	"AUTOMATIC"	"QP"	"Harvey"	-1	"TIME_LIMIT"
10	"optimizeCbModel"	"gurobi"	"PRIMAL"	"QP"	"Harvey"	-1	"TIME_LIMIT"
11	"optimizeCbModel"	"gurobi"	"DUAL"	"QP"	"Harvey"	-1	"TIME_LIMIT"
12	"optimizeCbModel"	"gurobi"	"BARRIER"	"QP"	"Harvey"	-1	"TIME_LIMIT"
13	"optimizeCbModel"	"mosek"	"FREE"	"LP"	"Harvey"	1	"OPTIMAL"
14	"optimizeCbModel"	"mosek"	"INTPNT"	"LP"	"Harvey"	1	"OPTIMAL"

:

1×12 table

interface	solver	method	problem	model	stat	origStat	time
"optimizeCbModel"	"ibm_cplex"	"AUTOMATIC"	"LP"	"Harvey"	1	"optimal"	17.237

1×12 table

interface	solver	method	problem	model	stat	origStat
"optimizeCbModel"	"ibm_cplex"	"PRIMAL"	"LP"	"Harvey"	3	"time limit exceeded"

1×12 table

interface	solver	method	problem	model	stat	origStat
"optimizeCbModel"	"ibm_cplex"	"DUAL"	"LP"	"Harvey"	3	"optimal with unscaled in"

1×12 table

interface	solver	method	problem	model	stat	origStat
"optimizeCbModel"	"ibm_cplex"	"NETWORK"	"LP"	"Harvey"	3	"optimal with unscaled"

1×12 table

interface	solver	method	problem	model	stat	origStat	time
"optimizeCbModel"	"ibm_cplex"	"BARRIER"	"LP"	"Harvey"	3	"optimal with unscaled"	1.000

```
"optimizeCbModel"    "ibm_cplex"      "BARRIER"       "LP"        "Harvey"       1      "optimal"     16.065
```

1x12 table

interface	solver	method	problem	model	stat	origStat	time
"optimizeCbModel"	"ibm_cplex"	"SIFTING"	"LP"	"Harvey"	1	"optimal"	100.04

1x12 table

interface	solver	method	problem	model	stat	origStat	time
"optimizeCbModel"	"ibm_cplex"	"CONCURRENT"	"LP"	"Harvey"	1	"optimal"	17.236

T = 25x12 table

	interface	solver	method	problem	model	stat	origStat
1	"optimizeCbModel"	"mosek"	"FREE"	"QP"	"Harvey"	1	"OPTIMAL"
2	"optimizeCbModel"	"mosek"	"INTPNT"	"QP"	"Harvey"	1	"OPTIMAL"
3	"optimizeCbModel"	"ibm_cplex"	"AUTOMATIC"	"QP"	"Harvey"	3	"non-optimal"
4	"optimizeCbModel"	"ibm_cplex"	"PRIMAL"	"QP"	"Harvey"	-1	"time limit exceeded"
5	"optimizeCbModel"	"ibm_cplex"	"DUAL"	"QP"	"Harvey"	-1	"time limit exceeded"
6	"optimizeCbModel"	"ibm_cplex"	"NETWORK"	"QP"	"Harvey"	-1	"time limit exceeded"
7	"optimizeCbModel"	"ibm_cplex"	"BARRIER"	"QP"	"Harvey"	3	"non-optimal"
8	"optimizeCbModel"	"ibm_cplex"	"CONCURRENT"	"QP"	"Harvey"	-1	"time limit exceeded"
9	"optimizeCbModel"	"gurobi"	"AUTOMATIC"	"QP"	"Harvey"	-1	"TIME_LIMIT"
10	"optimizeCbModel"	"gurobi"	"PRIMAL"	"QP"	"Harvey"	-1	"TIME_LIMIT"
11	"optimizeCbModel"	"gurobi"	"DUAL"	"QP"	"Harvey"	-1	"TIME_LIMIT"
12	"optimizeCbModel"	"gurobi"	"BARRIER"	"QP"	"Harvey"	-1	"TIME_LIMIT"
13	"optimizeCbModel"	"mosek"	"FREE"	"LP"	"Harvey"	1	"OPTIMAL"
14	"optimizeCbModel"	"mosek"	"INTPNT"	"LP"	"Harvey"	1	"OPTIMAL"
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮

1x12 table

interface	solver	method	problem	model	stat	origStat	time
"optimizeCbModel"	"gurobi"	"AUTOMATIC"	"LP"	"Harvey"	1	"OPTIMAL"	41.334

1x12 table

interface	solver	method	problem	model	stat	origStat	time
"optimizeCbModel"	"gurobi"	"PRIMAL"	"LP"	"Harvey"	-1	"TIME_LIMIT"	200.2

1×12 table

interface	solver	method	problem	model	stat	origStat	time
"optimizeCbModel"	"gurobi"	"DUAL"	"LP"	"Harvey"	-1	"TIME_LIMIT"	200.19

1×12 table

interface	solver	method	problem	model	stat	origStat	time
"optimizeCbModel"	"gurobi"	"BARRIER"	"LP"	"Harvey"	1	"OPTIMAL"	31.246

1×12 table

interface	solver	method	problem	model	stat	origStat	time
"optimizeCbModel"	"gurobi"	"CONCURRENT"	"LP"	"Harvey"	1	"OPTIMAL"	41.333

1×12 table

interface	solver	method	problem	model	stat	origStat
"optimizeCbModel"	"gurobi"	"DETERMINISTIC_CONCURRENT"	"LP"	"Harvey"	1	"OPTIMAL"

T = 31×12 table

	interface	solver	method	problem	model	stat	origStat
1	"optimizeCbModel"	"mosek"	"FREE"	"QP"	"Harvey"	1	"OPTIMAL"
2	"optimizeCbModel"	"mosek"	"INTPNT"	"QP"	"Harvey"	1	"OPTIMAL"
3	"optimizeCbModel"	"ibm_cplex"	"AUTOMATIC"	"QP"	"Harvey"	3	"non-optimal"
4	"optimizeCbModel"	"ibm_cplex"	"PRIMAL"	"QP"	"Harvey"	-1	"time limit exceeded"
5	"optimizeCbModel"	"ibm_cplex"	"DUAL"	"QP"	"Harvey"	-1	"time limit exceeded"
6	"optimizeCbModel"	"ibm_cplex"	"NETWORK"	"QP"	"Harvey"	-1	"time limit exceeded"
7	"optimizeCbModel"	"ibm_cplex"	"BARRIER"	"QP"	"Harvey"	3	"non-optimal"
8	"optimizeCbModel"	"ibm_cplex"	"CONCURRENT"	"QP"	"Harvey"	-1	"time limit exceeded"
9	"optimizeCbModel"	"gurobi"	"AUTOMATIC"	"QP"	"Harvey"	-1	"TIME_LIMIT"
10	"optimizeCbModel"	"gurobi"	"PRIMAL"	"QP"	"Harvey"	-1	"TIME_LIMIT"
11	"optimizeCbModel"	"gurobi"	"DUAL"	"QP"	"Harvey"	-1	"TIME_LIMIT"
12	"optimizeCbModel"	"gurobi"	"BARRIER"	"QP"	"Harvey"	-1	"TIME_LIMIT"
13	"optimizeCbModel"	"mosek"	"FREE"	"LP"	"Harvey"	1	"OPTIMAL"
14	"optimizeCbModel"	"mosek"	"INTPNT"	"LP"	"Harvey"	1	"OPTIMAL"

:

:

1×12 table

interface	solver	method	problem	model	stat	origStat	time	...
"optimizeCbModel"	"mosek"	"FREE"	"QP"	"Harvey"	1	"OPTIMAL"	61.866	{ [7.03
1x12 table								
interface	solver	method	problem	model	stat	origStat	time	...
"optimizeCbModel"	"mosek"	"INTPNT"	"QP"	"Harvey"	1	"OPTIMAL"	61.835	{ [7.
T = 33x12 table								
1	"optimizeCbModel"	"mosek"	"FREE"	"QP"	"Harvey"	1	"OPTIMAL"	
2	"optimizeCbModel"	"mosek"	"INTPNT"	"QP"	"Harvey"	1	"OPTIMAL"	
3	"optimizeCbModel"	"ibm_cplex"	"AUTOMATIC" "	"QP"	"Harvey"	3	"non-optimal"	
4	"optimizeCbModel"	"ibm_cplex"	"PRIMAL"	"QP"	"Harvey"	-1	"time limit exceeded"	
5	"optimizeCbModel"	"ibm_cplex"	"DUAL"	"QP"	"Harvey"	-1	"time limit exceeded"	
6	"optimizeCbModel"	"ibm_cplex"	"NETWORK"	"QP"	"Harvey"	-1	"time limit exceeded"	
7	"optimizeCbModel"	"ibm_cplex"	"BARRIER"	"QP"	"Harvey"	3	"non-optimal"	
8	"optimizeCbModel"	"ibm_cplex"	"CONCURRENT"	"QP"	"Harvey"	-1	"time limit exceeded"	
9	"optimizeCbModel"	"gurobi"	"AUTOMATIC" "	"QP"	"Harvey"	-1	"TIME_LIMIT"	
10	"optimizeCbModel"	"gurobi"	"PRIMAL"	"QP"	"Harvey"	-1	"TIME_LIMIT"	
11	"optimizeCbModel"	"gurobi"	"DUAL"	"QP"	"Harvey"	-1	"TIME_LIMIT"	
12	"optimizeCbModel"	"gurobi"	"BARRIER"	"QP"	"Harvey"	-1	"TIME_LIMIT"	
13	"optimizeCbModel"	"mosek"	"FREE"	"LP"	"Harvey"	1	"OPTIMAL"	
14	"optimizeCbModel"	"mosek"	"INTPNT"	"LP"	"Harvey"	1	"OPTIMAL"	
⋮								
1x12 table								
interface	solver	method	problem	model	stat	origStat	time	...
"optimizeCbModel"	"ibm_cplex"	"AUTOMATIC"	"QP"	"Harvey"	3	"non-optimal"	30.	
1x12 table								
interface	solver	method	problem	model	stat	origStat		
"optimizeCbModel"	"ibm_cplex"	"PRIMAL"	"QP"	"Harvey"	-1	"time limit exceeded"		
1x12 table								
interface	solver	method	problem	model	stat	origStat		

```
"optimizeCbModel"    "ibm_cplex"      "DUAL"        "QP"          "Harvey"       -1        "time limit exceeded"
```

1x12 table

interface	solver	method	problem	model	stat	origStat
"optimizeCbModel"	"ibm_cplex"	"NETWORK"	"QP"	"Harvey"	-1	"time limit exceeded"

1x12 table

interface	solver	method	problem	model	stat	origStat	time
"optimizeCbModel"	"ibm_cplex"	"BARRIER"	"QP"	"Harvey"	3	"non-optimal"	30.21

1x12 table

interface	solver	method	problem	model	stat	origStat
"optimizeCbModel"	"ibm_cplex"	"CONCURRENT"	"QP"	"Harvey"	-1	"time limit exceeded"

T = 39x12 table

	interface	solver	method	problem	model	stat	origStat
1	"optimizeCbModel"	"mosek"	"FREE"	"QP"	"Harvey"	1	"OPTIMAL"
2	"optimizeCbModel"	"mosek"	"INTPNT"	"QP"	"Harvey"	1	"OPTIMAL"
3	"optimizeCbModel"	"ibm_cplex"	"AUTOMATIC"	"QP"	"Harvey"	3	"non-optimal"
4	"optimizeCbModel"	"ibm_cplex"	"PRIMAL"	"QP"	"Harvey"	-1	"time limit exceeded"
5	"optimizeCbModel"	"ibm_cplex"	"DUAL"	"QP"	"Harvey"	-1	"time limit exceeded"
6	"optimizeCbModel"	"ibm_cplex"	"NETWORK"	"QP"	"Harvey"	-1	"time limit exceeded"
7	"optimizeCbModel"	"ibm_cplex"	"BARRIER"	"QP"	"Harvey"	3	"non-optimal"
8	"optimizeCbModel"	"ibm_cplex"	"CONCURRENT"	"QP"	"Harvey"	-1	"time limit exceeded"
9	"optimizeCbModel"	"gurobi"	"AUTOMATIC"	"QP"	"Harvey"	-1	"TIME_LIMIT"
10	"optimizeCbModel"	"gurobi"	"PRIMAL"	"QP"	"Harvey"	-1	"TIME_LIMIT"
11	"optimizeCbModel"	"gurobi"	"DUAL"	"QP"	"Harvey"	-1	"TIME_LIMIT"
12	"optimizeCbModel"	"gurobi"	"BARRIER"	"QP"	"Harvey"	-1	"TIME_LIMIT"
13	"optimizeCbModel"	"mosek"	"FREE"	"LP"	"Harvey"	1	"OPTIMAL"
14	"optimizeCbModel"	"mosek"	"INTPNT"	"LP"	"Harvey"	1	"OPTIMAL"
.							
.							
.							

1x12 table

interface	solver	method	problem	model	stat	origStat	time
"optimizeCbModel"	"gurobi"	"AUTOMATIC"	"QP"	"Harvey"	-1	"TIME_LIMIT"	201.39

1x12 table

interface	solver	method	problem	model	stat	origStat	time
"optimizeCbModel"	"gurobi"	"PRIMAL"	"QP"	"Harvey"	-1	"TIME_LIMIT"	200.21

1x12 table

interface	solver	method	problem	model	stat	origStat	time
"optimizeCbModel"	"gurobi"	"DUAL"	"QP"	"Harvey"	-1	"TIME_LIMIT"	200.18

1x12 table

interface	solver	method	problem	model	stat	origStat	time
"optimizeCbModel"	"gurobi"	"BARRIER"	"QP"	"Harvey"	-1	"TIME_LIMIT"	201

T = 43x12 table

	interface	solver	method	problem	model	stat	origStat
1	"optimizeCbModel"	"mosek"	"FREE"	"QP"	"Harvey"	1	"OPTIMAL"
2	"optimizeCbModel"	"mosek"	"INTPNT"	"QP"	"Harvey"	1	"OPTIMAL"
3	"optimizeCbModel"	"ibm_cplex"	"AUTOMATIC"	"QP"	"Harvey"	3	"non-optimal"
4	"optimizeCbModel"	"ibm_cplex"	"PRIMAL"	"QP"	"Harvey"	-1	"time limit exceeded"
5	"optimizeCbModel"	"ibm_cplex"	"DUAL"	"QP"	"Harvey"	-1	"time limit exceeded"
6	"optimizeCbModel"	"ibm_cplex"	"NETWORK"	"QP"	"Harvey"	-1	"time limit exceeded"
7	"optimizeCbModel"	"ibm_cplex"	"BARRIER"	"QP"	"Harvey"	3	"non-optimal"
8	"optimizeCbModel"	"ibm_cplex"	"CONCURRENT"	"QP"	"Harvey"	-1	"time limit exceeded"
9	"optimizeCbModel"	"gurobi"	"AUTOMATIC"	"QP"	"Harvey"	-1	"TIME_LIMIT"
10	"optimizeCbModel"	"gurobi"	"PRIMAL"	"QP"	"Harvey"	-1	"TIME_LIMIT"
11	"optimizeCbModel"	"gurobi"	"DUAL"	"QP"	"Harvey"	-1	"TIME_LIMIT"
12	"optimizeCbModel"	"gurobi"	"BARRIER"	"QP"	"Harvey"	-1	"TIME_LIMIT"
13	"optimizeCbModel"	"mosek"	"FREE"	"LP"	"Harvey"	1	"OPTIMAL"
14	"optimizeCbModel"	"mosek"	"INTPNT"	"LP"	"Harvey"	1	"OPTIMAL"
:							
:							

1x12 table

interface	solver	method	problem	model	stat	origStat	time	obj
"optimizeCbModel"	"mosek"	"FREE"	"LP"	"Harvey"	1	"OPTIMAL"	23.077	{ [2.65

1x12 table

interface	solver	method	problem	model	stat	origStat	time	obj
"optimizeCbModel"	"mosek"	"INTPNT"	"LP"	"Harvey"	1	"OPTIMAL"	23.255	{ [2.
OPTIMAL								
1x12 table								
interface	solver	method	problem	model	stat	origStat	time	obj
"optimizeCbModel"	"mosek"	"CONIC"	"LP"	"Harvey"	1	"OPTIMAL"	16.841	{ [2.
Mosek returned an error or warning, open the following link in your browser:								
https://docs.mosek.com/latest/toolbox/response-codes.html#mosek.rescode.trm_max_time								
1x12 table								
interface	solver	method	problem	model	stat			origStat
"optimizeCbModel"	"mosek"	"PRIMAL_SIMPLEX"	"LP"	"Harvey"	-1	"DUAL_FEASIBLE & MSK_RES_TOL"		
Mosek returned an error or warning, open the following link in your browser:								
https://docs.mosek.com/latest/toolbox/response-codes.html#mosek.rescode.trm_max_time								
1x12 table								
interface	solver	method	problem	model	stat			origStat
"optimizeCbModel"	"mosek"	"DUAL_SIMPLEX"	"LP"	"Harvey"	-1	"UNKNOWN & MSK_RES_TOL"		
Mosek returned an error or warning, open the following link in your browser:								
https://docs.mosek.com/latest/toolbox/response-codes.html#mosek.rescode.trm_max_time								
1x12 table								
interface	solver	method	problem	model	stat			origStat
"optimizeCbModel"	"mosek"	"FREE_SIMPLEX"	"LP"	"Harvey"	-1	"UNKNOWN & MSK_RES_TOL"		
T = 49x12 table								

	interface	solver	method	problem	model	stat	origStat
1	"optimizeCbModel"	"mosek"	"FREE"	"QP"	"Harvey"	1	"OPTIMAL"
2	"optimizeCbModel"	"mosek"	"INTPNT"	"QP"	"Harvey"	1	"OPTIMAL"
3	"optimizeCbModel"	"ibm_cplex"	"AUTOMATIC"	"QP"	"Harvey"	3	"non-optimal"
4	"optimizeCbModel"	"ibm_cplex"	"PRIMAL"	"QP"	"Harvey"	-1	"time limit exceeded"
5	"optimizeCbModel"	"ibm_cplex"	"DUAL"	"QP"	"Harvey"	-1	"time limit exceeded"
6	"optimizeCbModel"	"ibm_cplex"	"NETWORK"	"QP"	"Harvey"	-1	"time limit exceeded"
7	"optimizeCbModel"	"ibm_cplex"	"BARRIER"	"QP"	"Harvey"	3	"non-optimal"
8	"optimizeCbModel"	"ibm_cplex"	"CONCURRENT"	"QP"	"Harvey"	-1	"time limit exceeded"
9	"optimizeCbModel"	"gurobi"	"AUTOMATIC"	"QP"	"Harvey"	-1	"TIME_LIMIT"
10	"optimizeCbModel"	"gurobi"	"PRIMAL"	"QP"	"Harvey"	-1	"TIME_LIMIT"
11	"optimizeCbModel"	"gurobi"	"DUAL"	"QP"	"Harvey"	-1	"TIME_LIMIT"

	interface	solver	method	problem	model	stat	origStat
12	"optimizeCbModel"	"gurobi"	"BARRIER"	"QP"	"Harvey"	-1	"TIME_LIMIT"
13	"optimizeCbModel"	"mosek"	"FREE"	"LP"	"Harvey"	1	"OPTIMAL"
14	"optimizeCbModel"	"mosek"	"INTPNT"	"LP"	"Harvey"	1	"OPTIMAL"
	:						

1x12 table

interface	solver	method	problem	model	stat	origStat	time
"optimizeCbModel"	"ibm_cplex"	"AUTOMATIC"	"LP"	"Harvey"	1	"optimal"	17.153

1x12 table

interface	solver	method	problem	model	stat	origStat
"optimizeCbModel"	"ibm_cplex"	"PRIMAL"	"LP"	"Harvey"	3	"time limit exceeded"

1x12 table

interface	solver	method	problem	model	stat	origStat
"optimizeCbModel"	"ibm_cplex"	"DUAL"	"LP"	"Harvey"	3	"optimal with unscaled in"

1x12 table

interface	solver	method	problem	model	stat	origStat
"optimizeCbModel"	"ibm_cplex"	"NETWORK"	"LP"	"Harvey"	3	"optimal with unscaled"

1x12 table

interface	solver	method	problem	model	stat	origStat	time
"optimizeCbModel"	"ibm_cplex"	"BARRIER"	"LP"	"Harvey"	1	"optimal"	16.145

1x12 table

interface	solver	method	problem	model	stat	origStat	time
"optimizeCbModel"	"ibm_cplex"	"SIFTING"	"LP"	"Harvey"	1	"optimal"	99.974

1x12 table

interface	solver	method	problem	model	stat	origStat	time
"optimizeCbModel"	"ibm_cplex"	"CONCURRENT"	"LP"	"Harvey"	1	"optimal"	17.228

	interface	solver	method	problem	model	stat	origStat
1	"optimizeCbModel"	"mosek"	"FREE"	"QP"	"Harvey"	1	"OPTIMAL"

...

	interface	solver	method	problem	model	stat	origStat
2	"optimizeCbModel"	"mosek"	"INTPNT"	"QP"	"Harvey"	1	"OPTIMAL"
3	"optimizeCbModel"	"ibm_cplex"	"AUTOMATIC"	"QP"	"Harvey"	3	"non-optimal"
4	"optimizeCbModel"	"ibm_cplex"	"PRIMAL"	"QP"	"Harvey"	-1	"time limit exceeded"
5	"optimizeCbModel"	"ibm_cplex"	"DUAL"	"QP"	"Harvey"	-1	"time limit exceeded"
6	"optimizeCbModel"	"ibm_cplex"	"NETWORK"	"QP"	"Harvey"	-1	"time limit exceeded"
7	"optimizeCbModel"	"ibm_cplex"	"BARRIER"	"QP"	"Harvey"	3	"non-optimal"
8	"optimizeCbModel"	"ibm_cplex"	"CONCURRENT"	"QP"	"Harvey"	-1	"time limit exceeded"
9	"optimizeCbModel"	"gurobi"	"AUTOMATIC"	"QP"	"Harvey"	-1	"TIME_LIMIT"
10	"optimizeCbModel"	"gurobi"	"PRIMAL"	"QP"	"Harvey"	-1	"TIME_LIMIT"
11	"optimizeCbModel"	"gurobi"	"DUAL"	"QP"	"Harvey"	-1	"TIME_LIMIT"
12	"optimizeCbModel"	"gurobi"	"BARRIER"	"QP"	"Harvey"	-1	"TIME_LIMIT"
13	"optimizeCbModel"	"mosek"	"FREE"	"LP"	"Harvey"	1	"OPTIMAL"
14	"optimizeCbModel"	"mosek"	"INTPNT"	"LP"	"Harvey"	1	"OPTIMAL"

:

1×12 table

interface	solver	method	problem	model	stat	origStat	time
"optimizeCbModel"	"gurobi"	"AUTOMATIC"	"LP"	"Harvey"	1	"OPTIMAL"	41.581

1×12 table

interface	solver	method	problem	model	stat	origStat	time
"optimizeCbModel"	"gurobi"	"PRIMAL"	"LP"	"Harvey"	-1	"TIME_LIMIT"	200.19

1×12 table

interface	solver	method	problem	model	stat	origStat	time
"optimizeCbModel"	"gurobi"	"DUAL"	"LP"	"Harvey"	-1	"TIME_LIMIT"	200.19

1×12 table

interface	solver	method	problem	model	stat	origStat	time
"optimizeCbModel"	"gurobi"	"BARRIER"	"LP"	"Harvey"	1	"OPTIMAL"	30.577

1×12 table

interface	solver	method	problem	model	stat	origStat	time

```
"optimizeCbModel"    "gurobi"      "CONCURRENT"     "LP"        "Harvey"       1      "OPTIMAL"      42.292
```

1x12 table

interface	solver	method	problem	model	stat	origstat
"optimizeCbModel"	"gurobi"	"DETERMINISTIC_CONCURRENT"	"LP"	"Harvey"	1	"OPTIMAL"

T = 62x12 table

	interface	solver	method	problem	model	stat	origStat
1	"optimizeCbModel"	"mosek"	"FREE"	"QP"	"Harvey"	1	"OPTIMAL"
2	"optimizeCbModel"	"mosek"	"INTPNT"	"QP"	"Harvey"	1	"OPTIMAL"
3	"optimizeCbModel"	"ibm_cplex"	"AUTOMATIC"	"QP"	"Harvey"	3	"non-optimal"
4	"optimizeCbModel"	"ibm_cplex"	"PRIMAL"	"QP"	"Harvey"	-1	"time limit exceeded"
5	"optimizeCbModel"	"ibm_cplex"	"DUAL"	"QP"	"Harvey"	-1	"time limit exceeded"
6	"optimizeCbModel"	"ibm_cplex"	"NETWORK"	"QP"	"Harvey"	-1	"time limit exceeded"
7	"optimizeCbModel"	"ibm_cplex"	"BARRIER"	"QP"	"Harvey"	3	"non-optimal"
8	"optimizeCbModel"	"ibm_cplex"	"CONCURRENT"	"QP"	"Harvey"	-1	"time limit exceeded"
9	"optimizeCbModel"	"gurobi"	"AUTOMATIC"	"QP"	"Harvey"	-1	"TIME_LIMIT"
10	"optimizeCbModel"	"gurobi"	"PRIMAL"	"QP"	"Harvey"	-1	"TIME_LIMIT"
11	"optimizeCbModel"	"gurobi"	"DUAL"	"QP"	"Harvey"	-1	"TIME_LIMIT"
12	"optimizeCbModel"	"gurobi"	"BARRIER"	"QP"	"Harvey"	-1	"TIME_LIMIT"
13	"optimizeCbModel"	"mosek"	"FREE"	"LP"	"Harvey"	1	"OPTIMAL"
14	"optimizeCbModel"	"mosek"	"INTPNT"	"LP"	"Harvey"	1	"OPTIMAL"
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮

```
T.method = replace(T.method, 'MSK_OPTIMIZER_', '') ;
T.method = replace(T.method, '_', '') ;
T.solver = replace(T.solver, 'ibm_', '') ;
T.approach = append(T.solver, ' ', T.method) ;
T = sortrows(T, {'stat','time'}, {'ascend','ascend'}) ;
display(T)
```

T = 62x13 table

	interface	solver	method	problem	model	stat
1	"optimizeCbModel"	"gurobi"	"PRIMAL"	"QP"	"Harvey"	-1
2	"optimizeCbModel"	"gurobi"	"DUAL"	"QP"	"Harvey"	-1
3	"optimizeCbModel"	"gurobi"	"DUAL"	"QP"	"Harvey"	-1
4	"optimizeCbModel"	"gurobi"	"DUAL"	"LP"	"Harvey"	-1

	interface	solver	method	problem	model	stat
5	"optimizeCbModel"	"gurobi"	"DUAL"	"LP"	"Harvey"	-1
6	"optimizeCbModel"	"gurobi"	"PRIMAL"	"LP"	"Harvey"	-1
7	"optimizeCbModel"	"gurobi"	"PRIMAL"	"LP"	"Harvey"	-1
8	"optimizeCbModel"	"gurobi"	"BARRIER"	"QP"	"Harvey"	-1
9	"optimizeCbModel"	"gurobi"	"PRIMAL"	"QP"	"Harvey"	-1
10	"optimizeCbModel"	"mosek"	"PRIMAL SIMPLEX"	"LP"	"Harvey"	-1
11	"optimizeCbModel"	"gurobi"	"BARRIER"	"QP"	"Harvey"	-1
12	"optimizeCbModel"	"gurobi"	"AUTOMATIC"	"QP"	"Harvey"	-1
13	"optimizeCbModel"	"gurobi"	"AUTOMATIC"	"QP"	"Harvey"	-1
14	"optimizeCbModel"	"mosek"	"DUAL SIMPLEX"	"LP"	"Harvey"	-1
:						

```
save([resultsFolder 'results_benchmarkWBMsolvers.mat'], 'T')
```

```

if 1
    if 0
        % Create the first histogram
        histogram(T.time(T.stat==1 &
strcmp(T.problem,'LP')), 'NumBins', 100, 'FaceColor', 'r', 'FaceAlpha', 0.5); %
'r' sets the color to red
        hold on; % Keep the current plot so that the second histogram is
overlaid

        % Create the second histogram
        histogram(T.time(T.stat==1 &
strcmp(T.problem,'QP')), 'NumBins', 100, 'FaceColor', 'b', 'FaceAlpha', 0.5); %
'r' sets the color to red
        xlabel({'Whole body metabolic model LP solution time (seconds)', ...
[int2str(nMet) ' metabolites, ' int2str(nRxn) ' reactions.'] })
        ylabel('Number of solutions')
        title('Solution time depends on solver, method and problem');
        legend('LP', 'QP');
        hold off; % Release the hold for future plots

    else

        if ~exist('T0', 'var')
            T0 = T;
        else
            T = T0;
        end
    end
end

```

```

% Concatenate solver and method into 'approach'
T.approach = append(T.solver, ' ', T.method);
T = T(strcmp(T.problem,'LP') & T.stat==1,:);
% Calculate the mean solve time and standard deviation for each
approach
avg_times = varfun(@mean, T, 'InputVariables', 'time',
'GroupingVariables', 'approach');
std_times = varfun(@std, T, 'InputVariables', 'time',
'GroupingVariables', 'approach');

times = avg_times;
times.std_time = std_times.std_time;
% Sort both the avg_times and std_times by the mean solve time
[times, sort_idx] = sortrows(times, 'mean_time');

figure
% Create a bar plot with the sorted data
b = bar(times.mean_time, 'FaceColor', 'b', 'FaceAlpha', 0.5);
hold on;

% Add error bars using the sorted standard deviations
errorbar(times.mean_time, times.std_time, 'k', 'linestyle', 'none',
'LineWidth', 1.5);
xticks(1:length(times.approach))
xticklabels(times.approach)

% Add labels and title
xlabel('Approach', 'Interpreter', 'none');
ylabel('Solve Time (s)');
title('Successful LP solve times', 'Interpreter', 'none');

if size(T,1)>1
    figure
    % fastest times
    times = times(times.mean_time<mean(times.mean_time),:);
    % Create a bar plot with the sorted data
    b = bar(times.mean_time, 'FaceColor', 'b', 'FaceAlpha', 0.5);
    hold on;

    % Add error bars using the sorted standard deviations
    errorbar(times.mean_time, times.std_time, 'k', 'linestyle',
    'none', 'LineWidth', 1.5);
    xticks(1:length(times.approach))
    xticklabels(times.approach)

    % Add labels and title
    xlabel('Approach', 'Interpreter', 'none');
    ylabel('Solve Time (s)');

```

```

        title('Successful LP solve times (lowest 50%)', 'Interpreter',
'none');
    end

T = T0;
% Concatenate solver and method into 'approach'
T.approach = append(T.solver, ' ', T.method);
figure
T = T(strcmp(T.problem,'QP') & T.stat==1,:);
% Calculate the mean solve time and standard deviation for each
approach
avg_times = varfun(@mean, T, 'InputVariables', 'time',
'GroupingVariables', 'approach');
std_times = varfun(@std, T, 'InputVariables', 'time',
'GroupingVariables', 'approach');
times = avg_times;
times.std_time = std_times.std_time;
% Sort both the avg_times and std_times by the mean solve time
[times, sort_idx] = sortrows(times, 'mean_time');

% Create a bar plot with the sorted data
b = bar(times.mean_time, 'FaceColor', 'r', 'FaceAlpha', 0.5);
hold on;

% Add error bars using the sorted standard deviations
errorbar(times.mean_time, times.std_time, 'k', 'linestyle', 'none',
'LineWidth', 1.5);
xticks(1:length(times.approach))
xticklabels(times.approach)

% Add labels and title
xlabel('Approach', 'Interpreter', 'none');
ylabel('Solve Time (seconds)');
title('Successful QP solve times', 'Interpreter', 'none');

if size(T,1)>1
    figure
    % fastest times
    times = times(times.mean_time<mean(times.mean_time),:);
    % Create a bar plot with the sorted data
    b = bar(times.mean_time, 'FaceColor', 'r', 'FaceAlpha', 0.5);
    hold on;
    % Add error bars using the sorted standard deviations
    errorbar(times.mean_time, times.std_time, 'k', 'linestyle',
'none', 'LineWidth', 1.5);
    xticks(1:length(times.approach))
    xticklabels(times.approach)

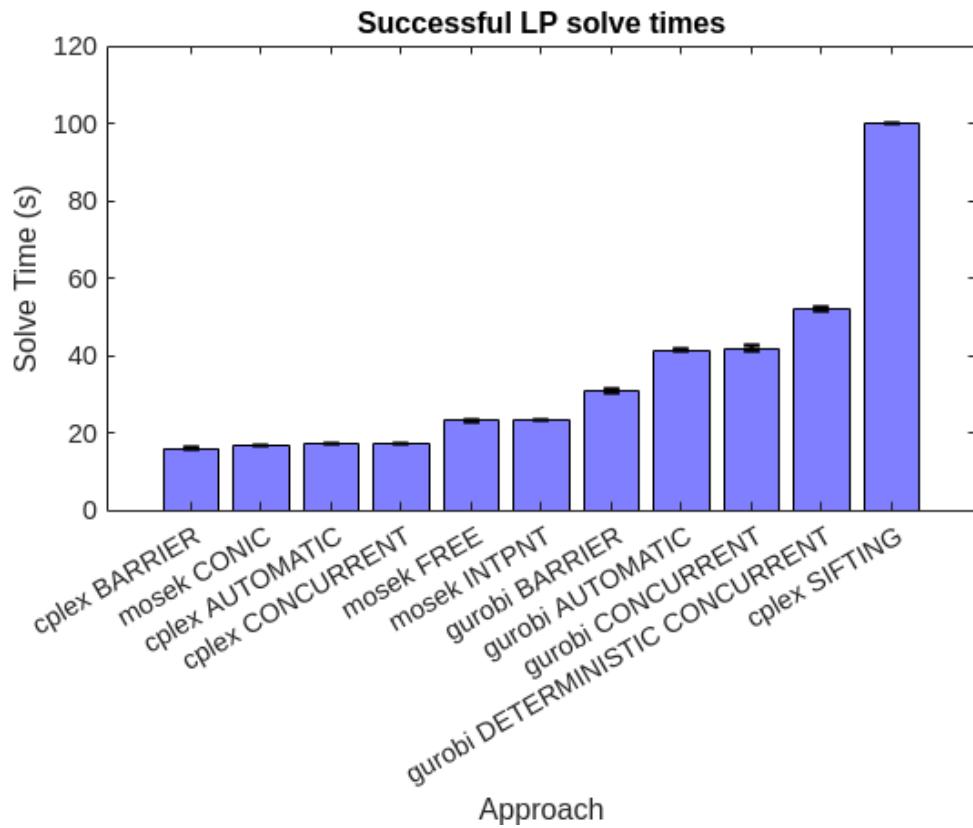
    % Add labels and title

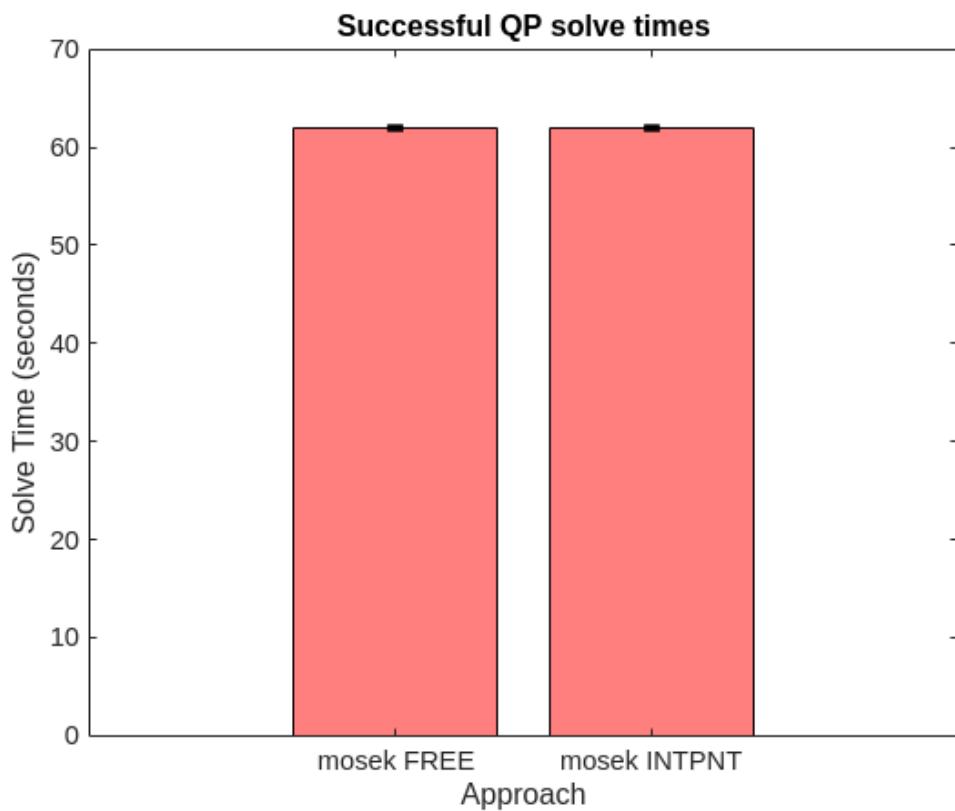
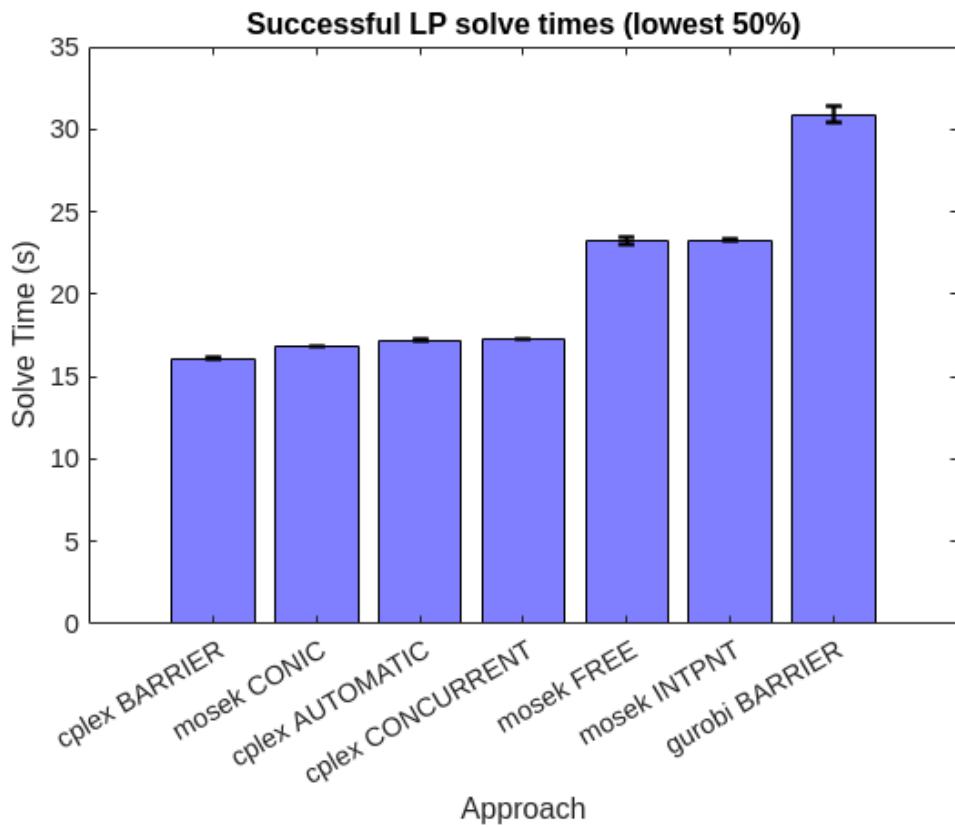
```

```

        xlabel('Approach', 'Interpreter', 'none');
        ylabel('Solve Time (seconds)');
        title('Successful QP solve times (lowest 50%)', 'Interpreter',
'none');
    end
end
end

```





Successful QP solve times (lowest 50%)

