Warm-up

(a) The Julia code is shown below. The result of using HiGHS, ECOS, and SCS solvers are respectively 80.0, 79.9999999936439, 80.00021048847219. They are slightly different because they have different ways to solve the problem, but they can be considered as the same results which nearly equal to 80.

(b) HiGHS is the fastest. Maybe it is because these three solvers solve the linear problem differently, and the HiGHS has better performance in this problem.The running times of using HiGHS, ECOS, and SCS solvers are respectively 0.000446, 0.001101, 0.001763(s).

(c) The threshold value of α is 8. The constraint is equivalent to $2x_1 - x_2 - 1/2x_3 \geq \alpha$, and the maximum value of $2x_1 - x_2 - 1/2x_3$ is 8 when $x_1 = 4, x_2 = x_3 = 0$, so when α is bigger than 8, the problem will be infeasible.

In [5]:
```julia
import Pkg
Pkg.add("HiGHS")
Pkg.add("JuMP")
Pkg.add("ECOS")
Pkg.add("SCS")
```

```
   Resolving package versions...
  No Changes to `C:\Users\X\.julia\environments\v1.8\Project.toml`
  No Changes to `C:\Users\X\.julia\environments\v1.8\Manifest.toml`
   Resolving package versions...
  No Changes to `C:\Users\X\.julia\environments\v1.8\Project.toml`
  No Changes to `C:\Users\X\.julia\environments\v1.8\Manifest.toml`
   Resolving package versions...
  No Changes to `C:\Users\X\.julia\environments\v1.8\Project.toml`
  No Changes to `C:\Users\X\.julia\environments\v1.8\Manifest.toml`
   Resolving package versions...
  No Changes to `C:\Users\X\.julia\environments\v1.8\Project.toml`
  No Changes to `C:\Users\X\.julia\environments\v1.8\Manifest.toml`
```

In [13]:
```julia
using JuMP

m = Model()

@variable(m, 0 <= x1 <= 4)
@variable(m, 0 <= x2 <= 4)
@variable(m, 0 <= x3 <= 4)
@constraint(m, x2 + x3/2 - 2x1 <= 0)
@objective(m, Max, 5x1 - x2 + 15x3)
```

Out[13]:

$$5x1 - x2 + 15x3$$

In [33]:
```julia
using HiGHS
set_optimizer(m, HiGHS.Optimizer)
@time optimize!(m)

println("The result is ", objective_value(m))
```

```
Running HiGHS 1.4.0 [date: 1970-01-01, git hash: bcf6c0b22]
Copyright (c) 2022 ERGO-Code under MIT licence terms
Presolving model
0 rows, 0 cols, 0 nonzeros
0 rows, 0 cols, 0 nonzeros
Presolve : Reductions: rows 0(-1); columns 0(-3); elements 0(-3) - Reduced to e
mpty
Solving the original LP from the solution after postsolve
Model   status      : Optimal
Objective value     :  8.0000000000e+01
HiGHS run time      :          0.00
  0.000446 seconds (189 allocations: 12.766 KiB)
The result is 80.0
```

In [30]:

```
using ECOS
set_optimizer(m, ECOS.Optimizer)
@time optimize!(m)

println("The result is ", objective_value(m))
```

```
  0.001101 seconds (996 allocations: 64.156 KiB)
The result is 79.9999999936439

ECOS 2.0.8 - (C) embotech GmbH, Zurich Switzerland, 2012-15. Web: www.embotech.
com/ECOS

It     pcost       dcost      gap   pres   dres    k/t    mu     step   sigma
IR    |   BT
 0  -3.807e+001  -1.499e+002  +1e+002  2e-003  3e-001  1e+000  1e+001    ---
---    1  1  - |  -  -
 1  -6.951e+001  -8.916e+001  +2e+001  3e-004  6e-002  1e+000  3e+000  0.9010
9e-002   0  0  0 |  0  0
 2  -7.790e+001  -9.311e+001  +1e+001  2e-004  6e-002  2e+000  2e+000  0.6129
4e-001   0  0  0 |  0  0
 3  -7.958e+001  -8.042e+001  +7e-001  1e-005  3e-003  1e-001  1e-001  0.9452
5e-004   0  0  0 |  0  0
 4  -8.000e+001  -8.001e+001  +9e-003  1e-007  4e-005  1e-003  1e-003  0.9881
3e-004   0  0  0 |  0  0
 5  -8.000e+001  -8.000e+001  +1e-004  2e-009  5e-007  2e-005  1e-005  0.9890
1e-004   1  0  0 |  0  0
 6  -8.000e+001  -8.000e+001  +1e-006  2e-011  5e-009  2e-007  1e-007  0.9890
1e-004   1  0  0 |  0  0
 7  -8.000e+001  -8.000e+001  +1e-008  2e-013  6e-011  2e-009  2e-009  0.9890
1e-004   1  0  0 |  0  0

OPTIMAL (within feastol=5.6e-011, reltol=1.5e-010, abstol=1.2e-008).
Runtime: 0.000068 seconds.
```

In [31]:

```
using SCS
set_optimizer(m, SCS.Optimizer)
@time optimize!(m)

println("The result is ", objective_value(m))
```

```
  0.001763 seconds (1.14 k allocations: 71.195 KiB)
The result is 80.00021048847219
------------------------------------------------------------------
               SCS v3.2.1 - Splitting Conic Solver
        (c) Brendan O'Donoghue, Stanford University, 2012
------------------------------------------------------------------
problem:  variables n: 3, constraints m: 7
cones:    l: linear vars: 7
settings: eps_abs: 1.0e-004, eps_rel: 1.0e-004, eps_infeas: 1.0e-007
          alpha: 1.50, scale: 1.00e-001, adaptive_scale: 1
          max_iters: 100000, normalize: 1, rho_x: 1.00e-006
          acceleration_lookback: 10, acceleration_interval: 10
lin-sys:  sparse-direct-amd-qdldl
          nnz(A): 9, nnz(P): 0
------------------------------------------------------------------
 iter | pri res | dua res |   gap    |   obj    |  scale   | time (s)
------------------------------------------------------------------
     0|1.49e+002 1.11e+000 2.38e+003 -1.27e+003 1.00e-001 6.09e-005
    75|1.52e-004 5.05e-005 5.32e-005 -8.00e+001 1.00e-001 9.58e-005
------------------------------------------------------------------
status:  solved
timings: total: 9.68e-005s = setup: 4.68e-005s + solve: 5.00e-005s
         lin-sys: 1.00e-005s, cones: 5.20e-006s, accel: 2.70e-006s
------------------------------------------------------------------
objective = -80.000184
------------------------------------------------------------------
```

Standard Form

(a) $x = (u, v, z_2, w)$, while $z_1 = u - v, z_3 = t - 5$. $A = \begin{bmatrix} 2 & -2 & -1 & 3 \\ -2 & 2 & 1 & -3 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$,

$b = \begin{bmatrix} 25 \\ -25 \\ 5 \\ 25 \end{bmatrix}, c = \begin{bmatrix} -3 \\ 3 \\ -2 \\ 1 \end{bmatrix}$.

(b) The code is written below.

In [4]:

```
using JuMP

m1 = Model()

@variable(m1, z1)
@variable(m1, 0 <= z2 <= 5)
@variable(m1, -5 <= z3 <= 20)
@constraint(m1, 2z1 - z2 + 3z3 == 10)
@objective(m1, Min, 3z1 + 2z2 - z3)

using HiGHS
set_optimizer(m1, HiGHS.Optimizer)
@time optimize!(m1)

println("The result is ", objective_value(m1))
```

```
Running HiGHS 1.4.0 [date: 1970-01-01, git hash: bcf6c0b22]
Copyright (c) 2022 ERGO-Code under MIT licence terms
Presolving model
0 rows, 2 cols, 0 nonzeros
0 rows, 0 cols, 0 nonzeros
Presolve : Reductions: rows 0(-1); columns 0(-3); elements 0(-3) - Reduced to e
mpty
Solving the original LP from the solution after postsolve
Model   status      : Optimal
Objective value     : -9.5000000000e+01
HiGHS run time      :          0.00
  0.000481 seconds (191 allocations: 12.984 KiB)
The result is -95.0
```

In [10]:

```
using JuMP

m2 = Model()

@variable(m2, u >= 0)
@variable(m2, v >= 0)
@variable(m2, z2 >= 0)
@variable(m2, w >= 0)
@constraint(m2, 2u - 2v - z2 + 3w <= 25)
@constraint(m2, -2u + 2v + z2 - 3w <= -25)
@constraint(m2, z2 <= 5)
@constraint(m2, w <= 25)
@objective(m2, Max, -3u + 3v - 2z2 + w)

using HiGHS
set_optimizer(m2, HiGHS.Optimizer)
@time optimize!(m2)

println("The result is ", - objective_value(m2) + 5) #original res = -(new res)
```

```
Running HiGHS 1.4.0 [date: 1970-01-01, git hash: bcf6c0b22]
Copyright (c) 2022 ERGO-Code under MIT licence terms
Presolving model
2 rows, 4 cols, 8 nonzeros
0 rows, 0 cols, 0 nonzeros
0 rows, 0 cols, 0 nonzeros
Presolve : Reductions: rows 0(-4); columns 0(-4); elements 0(-10) - Reduced to
empty
Solving the original LP from the solution after postsolve
Model   status      : Optimal
Objective value     :  1.0000000000e+02
HiGHS run time      :          0.00
  0.000503 seconds (185 allocations: 12.375 KiB)
The result is -95.0
```

Polyhedron modeling

There are five faces of the pyramid-shaped, and it can be defined by following inequalities:

$$z \geq 0$$

$$2y + z \leq 2$$

$$-2y + z \leq 2$$

$$2x + z \leq 2$$
$$-2x + z \leq 2$$

$$A = \begin{bmatrix} 0 & 2 & 1 \\ 0 & -2 & 1 \\ 2 & 0 & 1 \\ -2 & 0 & 1 \\ 0 & 0 & -1 \end{bmatrix}, b = \begin{bmatrix} 2 \\ 2 \\ 2 \\ 2 \\ 0 \end{bmatrix}.$$

Alloy blending problem

The company Steelco has received an order for 500 tonnes of steel to be used in shipbuilding. The steel must have the following charactersitics:

Chemical Element	Minimum Grade	Maximum Grade
Carbon (C)	2	3
Copper (Cu)	0.4	0.6
Manganese (Mn)	1.2	1.65

The company has seven different raw materials in stock that may be used for the production of this steel. The following table lists the grades, available amounts and prices for all materials:

Raw Material	C%	Cu%	Mn%	Availability in t	Cost in $/t
Iron1	2.5	0	1.3	400	200
Iron2	3	0	0.8	300	250
Iron3	0	0.3	0	600	150
Cu1	0	90	0	500	220
Cu2	0	96	4	200	240
Al1	0	0.4	1.2	300	200
Al2	0	0.6	0	250	165

The objective is to determine the composition of the steel that minimizes the production cost.

Problem data

In [13]:

```
using JuMP

raw = [:iron1, :iron2, :iron3, :cu1, :cu2, :al1, :al2]

# composition (in percent) of [C, Cu, Mn]
composition = Dict(
:iron1 => [2.5,0,1.3],
:iron2 => [3,0,0.8],
:iron3 => [0,0.3,0],
```

```julia
:cu1 => [0,90,0],
:cu2 => [0,96,4],
:al1 => [0,0.4,1.2],
:al2 => [0,0.6,0])

# availability in tonnes
availability = Dict(
:iron1 => 400,
:iron2 => 300,
:iron3 => 600,
:cu1 => 500,
:cu2 => 200,
:al1 => 300,
:al2 => 250)

# cost in dollars per tonne
cost = Dict(
:iron1 => 200,
:iron2 => 250,
:iron3 => 150,
:cu1 => 220,
:cu2 => 240,
:al1 => 200,
:al2 => 165)

# minimum and maximum grade of [C, Cu, Mn]
MinGrade = [2, 0.4, 1.2]
MaxGrade = [3, 0.6, 1.65]
;
```

In [32]:

```julia
using JuMP, HiGHS
m3 = Model()

@variable(m3, tons_raw[raw] >= 0 )

@expression(m3, tons_C, sum(tons_raw[i] * composition[i][1]/100 for i in raw) )
@expression(m3, tons_Cu,sum(tons_raw[i] * composition[i][2]/100 for i in raw) )
@expression(m3, tons_Mn,sum(tons_raw[i] * composition[i][3]/100 for i in raw) )
@expression(m3, totalcost,sum(tons_raw[i] * cost[i] for i in raw) )

@constraint(m3, sum(tons_raw[i] for i in raw) == 500 )
@constraint(m3, MinGrade[1]/100 <= tons_C/500 <= MaxGrade[1]/100)
@constraint(m3, MinGrade[2]/100 <= tons_Cu/500 <= MaxGrade[2]/100)
@constraint(m3, MinGrade[3]/100 <= tons_Mn/500 <= MaxGrade[3]/100)
for i in raw
    @constraint(m3, tons_raw[i] <= availability[i] )
end

@objective(m3, Min, totalcost)
```

Out[32]:

$$200 tons\_raw_{iron1} + 250 tons\_raw_{iron2} + 150 tons\_raw_{iron3} + 220 tons\_raw_{cu1} + 240 tons\_raw_{cu2} + 200 tons\_raw_{al1} + 165 tons\_raw_{al2}$$

In [31]:

```julia
set_optimizer(m3, HiGHS.Optimizer)
@time optimize!(m3)

println("The composition of the steel that minimize the production cost is:")
for i in raw
```

```
    println(value.(tons_raw[i]))
end
println("The result is ", objective_value(m3))
```

```
Running HiGHS 1.4.0 [date: 1970-01-01, git hash: bcf6c0b22]
Copyright (c) 2022 ERGO-Code under MIT licence terms
Presolving model
4 rows, 7 cols, 18 nonzeros
4 rows, 7 cols, 18 nonzeros
Presolve : Reductions: rows 4(-7); columns 7(-0); elements 18(-7)
Solving the presolved LP
Using EKK dual simplex solver - serial
  Iteration        Objective     Infeasibilities num(sum)
          0     0.0000000000e+00 Pr: 4(6.97825) 0s
          3     9.8121635792e+04 Pr: 0(0) 0s
Solving the original LP from the solution after postsolve
Model   status      : Optimal
Simplex   iterations: 3
Objective value     :  9.8121635792e+04
HiGHS run time      :          0.00
  0.000716 seconds (305 allocations: 20.766 KiB)
The composition of the steel that minimize the production cost is:
400.0
0.0
39.77630199231042
0.0
2.7612722824187346
57.46242572527084
0.0
The result is 98121.63579168124
```

The composition of the steel that minimize the production cost is: (as the order of :iron1, :iron2, :iron3, :cu1, :cu2, :al1, :al2) 400.0, 0.0, 39.77630199231042, 0.0, 2.7612722824187346, 57.46242572527084, 0.0 (tons). The result is 98121.63579168124($).

In [ ]:

```

```