# Wedding Planner

The job is to create a seating plan for all guests at a wedding. The data are for a real wedding, but the data are annonymized. The assignment is performed in 4 steps, which are then combined into a complete model. Notice, this is a rather hard problem to solve, the original wedding data had 74 guests, which is way to large to be solved directly, even for the commercial solvers.

## Problem 1

- Create a **feasible** table plan, such that all guests are seated at tables, not too many and couples are seated together. Hence we simple give a **constant** 42 as the objective

## Sets

- The different guest: $g \in Guests = \{1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20\}$
- The tables: $t \in Tables = \{1, 2, 3\}$

## Parameters

- $TableCap = 9$: The maximal number of guests sitting at each table
- $Couple_{g1,g2}$: Incidence matrixe, where $Couple_{g1,g2} = 1$ if guests $g1$ and $g2$ are a couple

## Decision variables

- Is guest $g$ seated at table $t$: $x_{g,t} \in \{0, 1\}$.

## Model

**Objective:**

- Only a feasible solution is needed (value chosen because of "The Hitchhiker's Guide to the Galaxy")

  42

**Constraints:**

- Ensure that all guests are seated:

$$\sum_t x_{g,t} = 1 \ \forall \ g$$

- Ensure that not too many guests are seated at each table:

$$\sum_g x_{g,t} \leq TableCap \ \forall \ t$$

- Ensure that couples are seated at the same table:

$$x_{g1,t} = x_{g2,t} \ \forall \ t, g1, g2 | Couple_{g1,g2} = 1$$

Notice the simple way couples are linked.

The full model in Julia/JuMP, available with the name

`WeddingPlanner1.jl`

from the book web-site, is given below:

```
#************************************************************************y
# WeddingPlanner Assignment 1, "Mathematical Programming Modelling" (42112)
using JuMP
using HiGHS
#************************************************************************y

#************************************************************************y
# PARAMETERS
include("WeddingData20.jl") # small dataset
println("Runing WeddingPlanner with $(G) guests, $(T) tables with capacity $(TableCap)")
#************************************************************************

#************************************************************************
# Model
wp =Model(HiGHS.Optimizer)

# 1 if guest g is sitting at table T
@variable(wp, x[g=1:G,t=1:T], Bin)
```

```
# Minimize dummy objective
@objective(wp, Min, 42 )

# all guests has to sit at a table
@constraint(wp, [g=1:G],
            sum( x[g,t] for t=1:T) == 1)

# dont exceed the number of persons at a table
@constraint(wp, [t=1:T],
            sum( x[g,t] for g=1:G) <= TableCap)

# couples should sit at the same table
@constraint(wp, [g1=1:G,g2=1:G,t=1:T; Couple[g1,g2]==1],
            x[g1,t] == x[g2,t])
#**************************************************************************

#**************************************************************************
# solve
optimize!(wp)
println("Termination status: $(termination_status(wp))")
#**************************************************************************

#**************************************************************************
# Report results
println("-------------------------------------");
if termination_status(wp) == MOI.OPTIMAL
    println("RESULTS:")
    println("objective = $(objective_value(wp))")
    for t = 1:T
        print("Table $(t) : ")
        for g1=1:G
            sh_g1=0
            if value(x[g1,t])==1
                print(" $(g1) ")
            end
        end
        println("")
        println("")
    end
else
  println("  No solution")
end
println("--------------------------------------");
#**************************************************************************y
```

Notice how the tableplan is printed out at the bottom of the file.

Given a feasible solution, the first job for you is to maximize the number of shared interests.

## Problem 2

- Maximize the number of shared interests for the table plan.

## New Parameters

- $SharedInterests_{g1,g2}$: The number of shared interests of guests $g1$ and $g2$

## New additional decision variables

- Are the guests $g1$ and $g2$ seated at table $t$: $y_{g1,g2,t} \in R^+$

## Model

**New objective:**

- Maximize the number of shared interests.

$$\sum_{g1,g2,t|g1<g2} SharedInterests_{g1,g2} \cdot y_{g1,g2,t}$$

**New additional constraints:**

- Ensure that $y_{g1,g2,t}$ can only take the value 1 if guests $g1$ sits at table $t$

$$y_{g1,g2,t} \leq x_{g1,t} \ \forall \ g1, g2, t|g1 < g2$$

- Ensure that $y_{g1,g2,t}$ can only take the value 1 if guests $g2$ sits at table $t$

$$y_{g1,g2,t} \leq x_{g2,t} \ \forall \ g1, g2, t|g1 < g2$$

The variable $y_{g1,g2,t}$ is used to liniarize the problem.

The full model in Julia/JuMP, available with the name

```
WeddingPlanner2.jl
```

from the book web-site, is given below:

```
#************************************************************************y
# WeddingPlanner Assignment 2, "Mathematical Programming Modelling" (42112)
using JuMP
using HiGHS
#************************************************************************y

#************************************************************************y
# PARAMETERS
include("WeddingData20.jl") # small dataset
println("Runing WeddingPlanner with $(G) guests, $(T) tables with capacity $(TableCap)")
#************************************************************************

#************************************************************************
# Model
wp =Model(HiGHS.Optimizer)

# 1 if guest g is sitting at table T
@variable(wp, x[g=1:G,t=1:T], Bin)

# 1 if guest g1 and guest g2 are both sitting at table T, symmetric
@variable(wp, 0 <= y[g1=1:G,g2=1:G,t=1:T] <= ( g1 < g2 ? 1 : 0) )

# Maximize the total amount of shared interests
@objective(wp, Max, sum( SharedInterests[g1,g2]*y[g1,g2,t]
                         for t=1:T, g1=1:G, g2=1:G if g1<g2) )

# all guests has to sit at a table
@constraint(wp, [g=1:G],
            sum( x[g,t] for t=1:T) == 1)

# dont exceed the number of persons at a table
@constraint(wp, [t=1:T],
            sum( x[g,t] for g=1:G) <= TableCap)

# couples should sit at the same table
@constraint(wp, [g1=1:G,g2=1:G,t=1:T; Couple[g1,g2]==1],
            x[g1,t] == x[g2,t])

# limit y, can only become 1 if g1 and g2 are at the same table
@constraint(wp, [g1=1:G,g2=1:G,t=1:T; g1<g2],
            y[g1,g2,t]  <= x[g1,t])

# limit y, can only become 1 if g1 and g2 are at the same table
```

```
@constraint(wp, [g1=1:G,g2=1:G,t=1:T; g1<g2],
            y[g1,g2,t]  <= x[g2,t])
#***************************************************************************

#***************************************************************************
# solve
optimize!(wp)
println("Termination status: $(termination_status(wp))")
#***************************************************************************

#***************************************************************************
# Report results
println("--------------------------------------");
if termination_status(wp) == MOI.OPTIMAL
    println("RESULTS:")
    println("objective = $(objective_value(wp))")
    for t = 1:T
        total_shared_int_table=0
        print("Table $(t) : ")
        for g1=1:G
            sh_g1=0
            if value(x[g1,t])==1
                print( " $(g1) ")
                for g2=1:G
                    if value(y[g1,g2,t])==1
                        sh_g1+=SharedInterests[g1,g2]
                    end
                end
                total_shared_int_table+=sh_g1
            end
        end
        println("")
        println("Penalties: (Shared interests: $( total_shared_int_table))")
    end
else
  println("  No solution")
end
println("---------------------------------------");
#***************************************************************************y


#***************************************************************************
println("Successfull end of $(PROGRAM_FILE)")
#***************************************************************************
```

Now, create a feasible table plan, such that the gender distribution becomes more equal, i.e. minimize the number of persons where there is either 3 more females than males or 3 more males than females.

## Problem 3

- Maximize the number of shared interests for the table plan.

## New Parameters

- $Male_g$: 1 if guest $g$ is a man
- $Female_g$: 1 if guest $g$ is a man

## New additional decision variables

- No missing males a table $t$: $m_t \geq 0$
- No missing females a table $t$: $f_t \geq 0$

## Model

**New objective:**

- Minimize the number of missing males or females

$$\sum_t (m_t + f_t)$$

**New additional constraints:**

- Set the number of missing males:

$$\sum_g (Male_g \cdot x_{g,t} - Female_g \cdot x_{g,t}) \leq 2 + m_t \ \forall \ t$$

- Set the number of missing females:

$$\sum_g (Female_g \cdot x_{g,t} - Male_g \cdot x_{g,t}) \leq 2 + m_t \ \forall \ t$$

Notice that the new variables $m_t$ and $f_t$ act as slack variables, which are forced to take a value greater that 0 if the difference between male and females is more than 2.

The full model in Julia/JuMP, available with the name

`WeddingPlanner3.jl`

from the book web-site, is given below:

```
#************************************************************************y
# WeddingPlanner Assignment 3, "Mathematical Programming Modelling" (42112)
using JuMP
using HiGHS
#************************************************************************y

#************************************************************************y
# PARAMETERS
include("WeddingData20.jl") # small dataset
println("Runing WeddingPlanner with $(G) guests, $(T) tables with capacity $(TableCap)")
#************************************************************************

#************************************************************************
# Model
wp = Model(HiGHS.Optimizer)

# 1 if guest g is sitting at table T
@variable(wp, x[g=1:G,t=1:T], Bin)

# excess males at table t
@variable(wp, m[t=1:T] >= 0)

# excess females at table t
@variable(wp, f[t=1:T] >= 0)

# Minimize the sum of males or females exceeding 2 at tables
@objective(wp, Min, sum( m[t] + f[t] for t=1:T))

# all guests has to sit at a table
@constraint(wp, [g=1:G],
            sum( x[g,t] for t=1:T) == 1)

# dont exceed the number of persons at a table
@constraint(wp, [t=1:T],
            sum( x[g,t] for g=1:G) <= TableCap)

# couples should sit at the same table
@constraint(wp, [g1=1:G,g2=1:G,t=1:T; Couple[g1,g2]==1],
            x[g1,t] == x[g2,t])

# if there are too many males at at table, penalty
@constraint(wp, [t=1:T],
            sum( Male[g]*x[g,t] - Female[g]*x[g,t] for g=1:G) <=
            m[t] + 2)

# if there are too many females at at table, penalty
@constraint(wp, [t=1:T],
            sum( Female[g]*x[g,t] - Male[g]*x[g,t] for g=1:G) <=
            f[t] + 2)
#************************************************************************

#************************************************************************
# solve
optimize!(wp)
println("Termination status: $(termination_status(wp))")
#************************************************************************

#************************************************************************
# Report results
println("-------------------------------------");
if termination_status(wp) == MOI.OPTIMAL
    println("RESULTS:")
    println("objective = $(objective_value(wp))")
    for t = 1:T
        total_shared_int_table=0
        print("Table $(t) : ")
        for g1=1:G
            sh_g1=0
            if value(x[g1,t])==1
                print(" $(g1) ")
            end
        end
        println("")
        println("Penalties: (Gender: $(value(m[t])+value(f[t])))")
    end
else
  println("  No solution")
end
println("--------------------------------------");
#*************************************************************************y
```

Now, create a feasible table plan, such that the number of people who do not know at least 3 other people at their table

## Problem 4

- Minimize the number of missing known persons, where each person should know at least 3

## New Parameters

- $Know_{g1,g2}$: 1 if guests $g1$ and guests $g2$ know each

## New additional decision variables

- The number of known other guests at the table for gust $g$: $k_g \geq 0$

## Model

**New objective:**

- Minimize the number of missing males or females

$$\sum_g k_g$$

**New additional constraints:**

- Force the "missing known other guests" for each guest:

$$k_g - 3 \cdot x_{g,t} + \sum_{g1} Know_{g,g1} \cdot (y_{g,g1,t} + y_{g1,g,t}) \geq 0 \;\forall\; g,t$$

The full model in Julia/JuMP, available with the name

`WeddingPlanner4.jl`

from the book web-site, is given below:

```
#************************************************************************
# WeddingPlanner Assignment 4, "Mathematical Programming Modelling" (42112)
using JuMP
using HiGHS
#************************************************************************y

#************************************************************************y
# PARAMETERS
```

```julia
include("WeddingData20.jl") # small dataset
println("Runing WeddingPlanner with $(G) guests, $(T) tables with capacity $(TableCap)")
#**************************************************************************

#**************************************************************************
# Model
wp =Model(HiGHS.Optimizer)

# 1 if guest g is sitting at table T
@variable(wp, x[g=1:G,t=1:T], Bin)

# 1 if guest g1 and guest g2 are both sitting at table T, symmetric
@variable(wp, 0 <= y[g1=1:G,g2=1:G,t=1:T] <= ( g1 < g2 ? 1 : 0) )

# guest g is missing known people
@variable(wp, k[g=1:G] >= 0)

# Minimize the number of people at tables where they know less than 3
@objective(wp, Min,sum( k[g] for g=1:G))

# all guests has to sit at a table
@constraint(wp, [g=1:G],
            sum( x[g,t] for t=1:T) == 1)

# dont exceed the number of persons at a table
@constraint(wp, [t=1:T],
            sum( x[g,t] for g=1:G) <= TableCap)

# couples should sit at the same table
@constraint(wp, [g1=1:G,g2=1:G,t=1:T; Couple[g1,g2]==1],
            x[g1,t] == x[g2,t])

# limit y, can only become 1 if g1 and g2 are at the same table
@constraint(wp, [g1=1:G,g2=1:G,t=1:T; g1<g2],
            y[g1,g2,t]  <= x[g1,t])

# limit y, can only become 1 if g1 and g2 are at the same table
@constraint(wp, [g1=1:G,g2=1:G,t=1:T; g1<g2],
            y[g1,g2,t]  <= x[g2,t])

# if a person sits at a table, penalty if not enough known people
@constraint(wp, [g=1:G,t=1:T],
            k[g] - 3*x[g,t]
            + sum( Know[g,g1]*(y[g,g1,t]+y[g1,g,t]) for g1=1:G) >= 0)
#**************************************************************************
```

```
#**************************************************************************
# solve
optimize!(wp)
println("Termination status: $(termination_status(wp))")
#**************************************************************************

#**************************************************************************
# Report results
println("-------------------------------------");
if termination_status(wp) == MOI.OPTIMAL
    println("RESULTS:")
    println("objective = $(objective_value(wp))")
    for t = 1:T
        total_not_know_at_table=0
        print("Table $(t) : ")
        for g1=1:G
            sh_g1=0
            if value(x[g1,t])==1
                total_not_know_at_table+=value(k[g1])
                print(" $(g1) ")
            end
        end
        println("")
        println("Penalties: (Know: $(total_not_know_at_table))")
    end
else
  println("  No solution")
end
println("--------------------------------------");
#**************************************************************************y
```

Finally, put everything together, minizing the sum of **negative shared interests** plus the missing males and females plus the number of missing knowns at tables.

## Problem 5

- Minimzie all.

## Model

**New objective:**

- Minimize the number of missing males or females

$$-\sum_{g1,g2,t|g1<g2} SharedInterests_{g1,g2} \cdot y_{g1,g2,t}+$$

$$\sum_t (m_t + f_t)+$$

$$\sum_g k_g$$

The full model in Julia/JuMP, available with the name

`WeddingPlanner5.jl`

from the book web-site, is given below:

```
#**************************************************************************y
# WeddingPlanner Assignment5 , "Mathematical Programming Modelling" (42112)
using JuMP
using HiGHS
#**************************************************************************y

#**************************************************************************y
# PARAMETERS
include("WeddingData20.jl") # small dataset
println("Runing WeddingPlanner with $(G) guests, $(T) tables with capacity $(TableCap)")
#*************************************************************************

#*************************************************************************
# Model
wp =Model(HiGHS.Optimizer)

# 1 if guest g is sitting at table T
@variable(wp, x[g=1:G,t=1:T], Bin)

# 1 if guest g1 and guest g2 are both sitting at table T, symmetric
@variable(wp, 0 <= y[g1=1:G,g2=1:G,t=1:T] <= ( g1 < g2 ? 1 : 0) )

# excess males at table t
@variable(wp, m[t=1:T] >= 0)

# excess females at table t
@variable(wp, f[t=1:T] >= 0)

# guest g is missing known people
@variable(wp, k[g=1:G] >= 0)
```

```
# Minimize sum of different objectives
@objective(wp, Min,
           # minimize the negative total shared intersts
           - sum( SharedInterests[g1,g2]*y[g1,g2,t]
                  for t=1:T, g1=1:G, g2=1:G if g1<g2)

           # minimize unbalanced male and females
           + sum( m[t] + f[t] for t=1:T)

           # minimize the no of guests not knowing enough at the table
           + sum( k[g] for g=1:G))

# all guests has to sit at a table
@constraint(wp, [g=1:G],
            sum( x[g,t] for t=1:T) == 1)

# dont exceed the number of persons at a table
@constraint(wp, [t=1:T],
            sum( x[g,t] for g=1:G) <= TableCap)

# couples should sit at the same table
@constraint(wp, [g1=1:G,g2=1:G,t=1:T; Couple[g1,g2]==1],
            x[g1,t] == x[g2,t])

# if there are too many males at at table, penalty
@constraint(wp, [t=1:T],
            sum( Male[g]*x[g,t] - Female[g]*x[g,t] for g=1:G) <=
            m[t] + 2)

# if there are too many females at at table, penalty
@constraint(wp, [t=1:T],
            sum( Female[g]*x[g,t] - Male[g]*x[g,t] for g=1:G) <=
            f[t] + 2)

# limit y, can only become 1 if g1 and g2 are at the same table
@constraint(wp, [g1=1:G,g2=1:G,t=1:T; g1<g2],
            y[g1,g2,t]  <= x[g1,t])

# limit y, can only become 1 if g1 and g2 are at the same table
@constraint(wp, [g1=1:G,g2=1:G,t=1:T; g1<g2],
            y[g1,g2,t]  <= x[g2,t])

# if a person sits at a table, penalty if not enough known people
@constraint(wp, [g=1:G,t=1:T],
            k[g] - 3*x[g,t] +
            sum( Know[g,g1]*(y[g,g1,t]+y[g1,g,t]) for g1=1:G) >= 0)
#**************************************************************************

#**************************************************************************
# solve
optimize!(wp)
println("Termination status: $(termination_status(wp))")
#**************************************************************************

#**************************************************************************
# Report results
println("----------------------------------------");
if termination_status(wp) == MOI.OPTIMAL
    println("RESULTS:")
    println("objective = $(objective_value(wp))")
    for t = 1:T
        total_shared_int_table=0
        total_not_know_at_table=0
        print("Table $(t) : ")
        for g1=1:G
            sh_g1=0
            if value(x[g1,t])==1
                total_not_know_at_table+=value(k[g1])
                print(" $(g1) ")
                for g2=1:G
                    if value(y[g1,g2,t])==1
                        sh_g1+=SharedInterests[g1,g2]
                    end
                end
                total_shared_int_table+=sh_g1
            end
        end
        println("")
        println("\nPenalties: (Shared interests: $(-total_shared_int_table), Gender: $((valu
    end
else
  println("  No solution")
end
println("-----------------------------------------");
#**************************************************************************y
```