Epsilon School Expansion

Aarushi, Anshu, Raihan

Problem

- A new wing is being added to the Epsilon School of Mathematics and Science.
- The number of incoming students is increasing by 140 over the graduating senior class.
- Due to the disparity of teacher distribution on campus, 7 new teachers can be hired.
- What department should each new teacher go to? Should each department get 1 teacher, or should the new teachers be hired according to the number of students taking each class?

Assumptions

- Teachers can teach 1 class at a time.
- Language teachers can choose to teach 2 languages, or only one language.
- Some classes are harder/easier to teach. This means the class will be weighted more on a scale of effort required by the teacher.
- Students can dual-enroll in departments/subjects, such as math, science, etc.
- Students cannot drop out or switch classes during the school year.

Important Variables

- Distribution of students enrolled in classes
- # students enrolled last year, # new students
- Dropout rate
- Current # of teachers
- Total possible number of teachers
- Variance in teacher-student ratio

Analysis of Problem

- Objective:
  - Students per teacher somewhat equal across teachers.
  - Equity – harder classes to teach should have less students.
  - Fire & hire teachers for fairest arrangement.
- We treat this as an optimization problem with constraints and objective
  - Variables (number of teachers) are integers
  - Constraint: Total number of teachers <=38
  - Objective: Minimize the variance in effort per teacher

Number of students enrolled

- Big assumption: Same course distributions year to year
- # of 10th gr. enrolled in each course = that of last year multiplied by the ratio of number of students.
- Effort for teaching each subject is different
- 5% percent dropout / yr, and students increased from 490 to 640
- Assumption: Only can drop out between years, nobody joins mid-yr.

Departmental Enrollment Totals: September 2024

Department	10th	11th	12th	Total
Art	31	33	35	99
Biology	198	95	26	319
Chemistry	59	126	109	294
English	183	155	152	490
French	41	32	49	122
German	19	22	10	51
Spanish	51	26	33	110
Mathematics	184	201	262	647
Music	50	56	49	155
Physics	50	58	183	291
Social Studies	183	131	59	373

$$x + 0.95x + 0.95(0.95x) = 490$$

- x = # 10th grade
- Normal years avg students (add 140):

10th grade	11th grade	12th grade
171.78	163.19	155.03

Model v1

- Excel spreadsheet

Next year enrollment										
	10th	11th	12th	Total		Faculty Dept	# of Teachers	Effort weights	students per teacher	students per teacher (weighted)	Standard dev.
Art	55.31736527	33	35	123.3173653		Art	1	1	123.3173653	123.3173653	
Biology	353.3173653	95	26	474.3173653		Biology	5	1.3	94.86347305	123.322515	
Chemistry	105.2814371	126	109	340.2814371		Chemistry	4	1.3	85.07035928	110.5914671	
English	326.5508982	155	152	633.5508982		English	5	1	126.7101796	126.7101796	
French	73.16167665	32	49	154.1616766		French	2	1.2	77.08083832	92.49700599	
German	33.90419162	22	10	65.90419162		German	1	1.2	65.90419162	79.08502994	
Spanish	91.00598802	26	33	150.005988		Spanish	2	1.2	75.00299401	90.00359281	
Math	328.3353293	201	262	791.3353293		Math	8	1.3	98.91691617	128.591991	
Music	89.22155689	56	49	194.2215569		Music	2	1	97.11077844	97.11077844	
Physics	89.22155689	58	183	330.2215569		Physics	4	1.3	82.55538922	107.322006	
Social Studies	326.5508982	131	59	516.5508982		Social Studies	4	1	129.1377246	129.1377246	
							38		95.97001905	109.7899687	17.83564158

- GRG Nonlinear Solver to minimize std. dev.
- Constraints: 38 teachers, 1+ teachers for each subject
- Problems

Excel Results & Issues

Issues with this approach:

- Extremely slow
- Less control
- Language teachers
- Not applicable to other scenarios

Subject	# of teachers
Art	1
Biology	5
Chemistry	4
English	5
French	2
German	1
Spanish	2
Math	8
Music	2
Physics	4
Social Studies	4

Exploring Other Choices:

- Needs to be fast
- Be able to run various scenarios
- Adaptable constraints/variables
- Julia with HiGHS optimizer

Model v2

EpsilonSchool.ipynb | Untitled.ipynb

Code | Share | Notebook | Julia 1.10.5

```julia
import MathOptInterface as MOI
import CSV, DataFrames, HiGHS
using HiGHS
C = CSV.read("EpsilonDatasheet.csv", DataFrames.DataFrame)
NumDept = size(C, 1)
display(C)
#display(NumDept)

#define the model
model = Model(HiGHS.Optimizer)


#define variables
@variable(model, 0 <= x[i=1:NumDept], Int )
NumOfExistingTeacher =38

#define constraints
@constraint(model, constrainttotal, 38 >= sum(x[i] for i in 1:NumDept) >= 38)

Inverse_Average_Weighted_Effort = NumOfExistingTeacher/sum(C[:, 2])

@expression(
    model,
    y[i=1:NumDept],
    x[i]/C[i, 2] - Inverse_Average_Weighted_Effort
)


# To compute t >= sum | y_i |, use NormOneCone, see:
# https://jump.dev/JuMP.jl/stable/tutorials/linear/tips_and_tricks/#L1-norm
@variable(model, t >= 0)
@constraint(model, [t; y] in MOI.NormOneCone(1 + length(y)))
@objective(model, Min, t)



# Printing the prepared optimization model
#set_silent(model)
print(model)

# Printing the prepared optimization model
g = open("example_model.txt", "w")
println(g, model)
close(g)
```

```julia
# Printing the prepared optimization model
#set_silent(model)
print(model)

# Printing the prepared optimization model
g = open("example_model.txt", "w")
println(g, model)
close(g)


# Solving the optimization problem
JuMP.optimize!(model)

# Printing the optimal solutions obtained
println("Optimal Solutions:")
for i in 1:NumDept
  println(C[i,1]," = ", JuMP.value(x[i]))
end
```

11×2 DataFrame

Row	Subject	Total
	String15	Float64
1	Art	123.317
2	Biology	616.613
3	Chemistry	442.366
4	English	633.551
5	French	184.994
6	German	79.085
7	Spanish	180.007
8	Math	1028.74
9	Music	194.222
10	Physics	429.288
11	Social Studies	516.551

$$\min \quad t$$

- Model is defined using JuMP (Julia's package for optimization) with the HiGHS optimizer, a solver for linear and mixed-integer programming problems.

Model v2 – cont.

- Datasheet : "EpsilonDatasheet.csv” contains effort per department
- x[i]: the number of new teachers allocated to each department i
- t: sum of absolute deviation in effort across departments
- Objective : minimize t
- Constraints
  - Each x is integer
  - Sum of x’s is less than or equal to 38

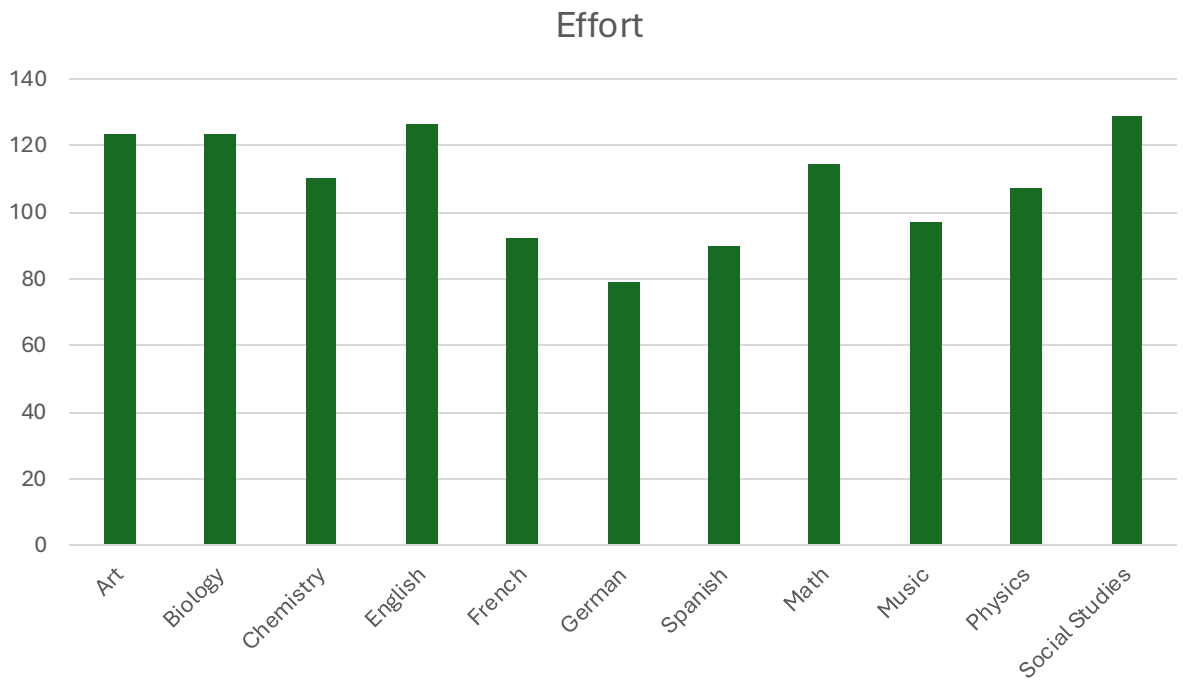
Effort Calculation:

- **Inverse_Average_Weighted_Effort**: total # of current teachers / the sum of students in each department
- y[i]: deviation of each department's effort from the inverse average weighted effort

Minimizing Effort Variance:

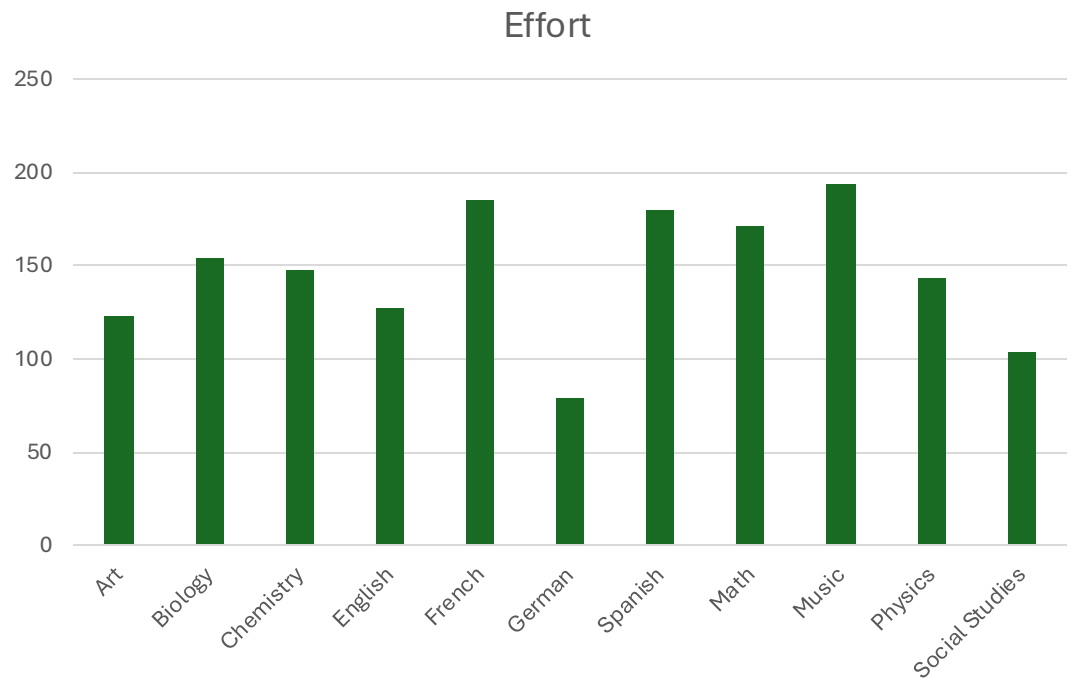
- Model uses a NormOneCone constraint to compute the L1 norm of the effort deviations (y[i]), meaning the sum of absolute deviations is minimized.
- Distribute new teachers so that the difference in workload (effort) among departments is minimized.

Effort comparisons



Solution

Standard Deviation: 16.8



Previous

Standard Deviation: 35.9

Solution Process

1. Model Setup and Constraints: The model sets up constraints for new teacher distribution and calculates the deviation in teaching effort across departments.

1.Optimization: JuMP's optimizer is used to solve the problem and find the optimal distribution of teachers that minimizes the total effort deviation.

2. Output: After solving, the model prints the optimal number of new teachers to assign to each department, balancing student numbers and teacher effort.

How it Solves the Problem:
The model ensures **fairness** by minimizing the difference in effort (students per teacher) across departments.

By using the L1 norm, the model **reduces the maximum workload variance** across departments.

It provides an optimal distribution that **aligns teacher allocations with student demand**, ensuring departments with higher student numbers receive more attention.

Solution

- The solution provided by the optimization model within the constraints was:
  - Art: 1
  - Biology: 5
  - Chemistry: 4
  - English: 5
  - French: 2
  - German: 1 (shared from spanish)
  - Spanish: 2 (one also teaches German)
  - Math: 9
  - Music: 2
  - Physics: 4
  - Social Studies: 4
- This solution makes the efforts for each department as close to each other as possible.

Strengths/Weaknesses

Strengths

- Foreign language teachers
- Model was realistic
- Within constraints
- Will work for multiple problems, can put many values in/many numbers of students

Weaknesses

- Middle of the year dropouts
- Only applies to next year
- Unrealistic assumptions about firing/hiring
- Outliers do not affect the standard deviation significantly

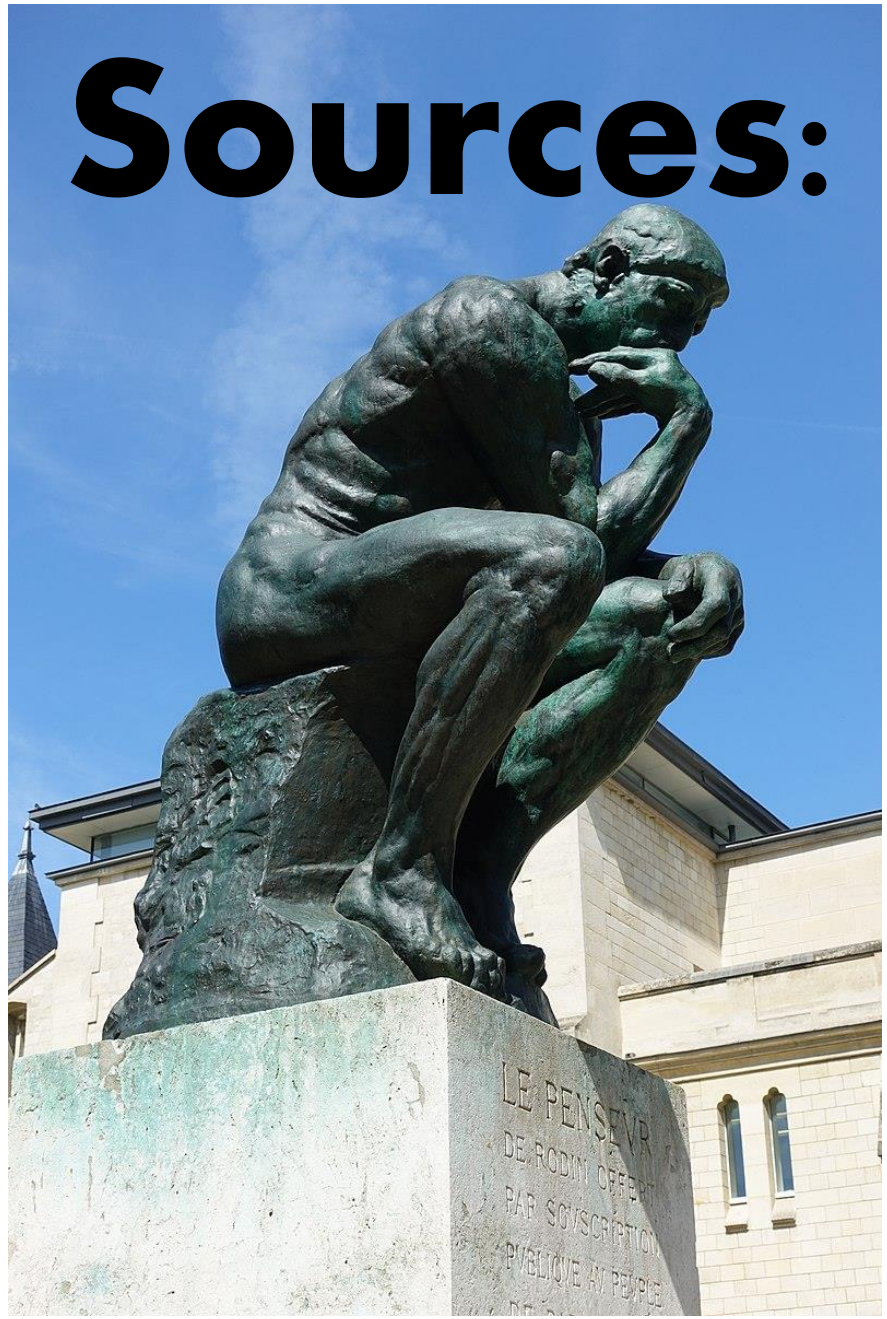
Future Steps

- Run the model on the next two years (when the incoming class graduates)
- Account for drop-outs or class changes during the year
- Test changes in model if teachers were to retire during the year
- Model a formula that can be used in different scenarios/ schools with different variables

Thank You

Acknowledgements:

- Excel
- Julia
- Mrs. Burns



Reference for Model

Min t; Subject to

constrainttotal : x[1] + x[2] + x[3] + x[4] + x[5] + x[6] + x[7] + x[8] + x[9] + x[10] + x[11] ∈ [38, 38]

[t, 0.00810915800517836 6 x[1] - 0.0085803392063282 89, 0.0016217638768754859 x[2] – 0.0085803392063282 89, 0.0022605722364363915 x[3] - 0.0085803392063282 89, 0.0015784051492013 18 x[4] - 0.0085803392063282 89, 0.0054055803738153 32 x[5] - 0.0085803392063282 89, 0.0126446180871231 52 x[6] - 0.0085803392063282 89, 0.0055553337866307 93 x[7] - 0.0085803392063282 89, 0.0009720667595853617 x[8] - 0.0085803392063282 89, 0.0051487590562095 84 x[9] - 0.0085803392063282 89, 0.0023294383819102 3 x[10] - 0.0085803392063282 89, 0.0019359176481633306 x[11] - 0.0085803392063282 89] ∈ MathOptInterface.NormOneCone(12)

x[1] ≥ 0 , x[2] ≥ 0, x[3] ≥ 0, x[4] ≥ 0, x[5] ≥ 0, x[6] ≥ 0, x[7] ≥ 0, x[8] ≥ 0, x[9] ≥ 0, x[10] ≥ 0, x[11] ≥ 0, t ≥ 0

x[1], x[2], x[3], x[4], x[5], x[6], x[7], x[8], x[9], x[10], x[11] ∈ Integers