Optimality cuts

```
using JuMP , GLPK , NLPModels , NLPModelsJuMP
```

```
using DataFrames
```

general type

$$\theta \geq e_l - E_l x$$

$$e = \sum_{i=1}^{K} p_i \lambda_i h_i$$

$$E = \sum_{i=1}^{K} p_i \lambda_i h_i$$

- p comes from problem description
- λ comes from solving the sub problem
- h and T are coming from sub problem with no solution

Example

Stage 1

$$\min 100x_1 + 150x_2$$

$$x_1 + x_2 \leq 120$$

$$-x_1 \leq -40$$

$$-x_2 \leq -20$$

```
[40.0, 20.0]
```

```
begin
    c1 = [100 ; 150]
    A1 = [1 1 ; -1 0 ; 0 -1]
    b1 = [120 ; -40 ; -20]
    main = Model(GLPK.Optimizer)
    @variable(main, x[1:2])
    @variable(main, -1000 ≤ θ)
    @objective(main, Min, c1' * x + θ)
    @constraint(main, A1 * x .≤ b1)
    optimize!(main)
    x = value.(x)
end
```

sub problem 1

$$\min -24y_1 - 28y_2$$

$$6y_1 + 10y_2 - 60x_1 \leq 0$$

$$8y_1 + 5y_2 - 80x_2 \leq 0$$

$$y_1 \leq 500$$

$$y_2 \leq 100$$

sub problem 2

$$\min -28y_1 - 32y_2$$

$$6y_1 + 10y_2 - 60x_1 \leq 0$$

$$8y_1 + 5y_2 - 80x_2 \leq 0$$

$$y_1 \leq 300$$

$$y_2 \leq 300$$

```
2×4 Matrix{Int64}:
 0  0  500  100
 0  0  300  300
```

```
begin
    I = 2
    c2 = [-24 -28 ; -28 -32];
    A2 = [6 10 ; 8 5 ; 1 0 ; 0 1];
    A3 = [-60 0 ; 0 -80 ; 0 0 ; 0 0];
    b2 = [0 0 500 100 ; 0 0 300 300];
end
```

```
solve_sub (generic function with 1 method)
```

```
function solve_sub()
    λ = zeros(2,4)
    for i in 1:I
        sub = Model(GLPK.Optimizer)
        @variable(sub, 0 ≤ y[1:2])
        @objective(sub, Min, c2[i , :]' * y)
        @constraint(sub, A2 * y + A3 * x .≤ b2[i , :])
        optimize!(sub)
        all_cons = all_constraints(sub, AffExpr, MOI.LessThan{Float64})
        λ[i , :] = dual.(all_cons)
    end
    return λ
end
```

```
2×4 Matrix{Float64}:
  0.0   -3.0   0.0  -13.0
 -2.32  -1.76  0.0    0.0
```

```
solve_sub()
```

```
no_solve_sub = no_solve_sub (generic function with 1 method)
```

```
no_solve_sub = function no_solve_sub()
    T = zeros(2,4)
    h = zeros(2,4)
    for i in 1:I
        sub = Model(GLPK.Optimizer)
        @variable(sub, x[1:2])
        @variable(sub, 0 ≤ y[1:2])
        @objective(sub, Min, c2[i , :]' * y)
        @constraint(sub, A2 * y + A3 * x .≤ b2[i , :])
        nlp = MathOptNLPModel(sub)
        l = zeros(nlp.meta.nvar)
        h[i , :] = nlp.meta.ucon
        #all_var = all_variables(sub)
        #var_index = [all_var[s].index.value for s in 1:length(all_var)]
        #df = DataFrame(varName = all_var , varIndex = var_index)
        T = Matrix(jac(nlp,l))[ : , 1:2]
    end
    return (T, h, all_var)
end
```

```
UndefVarError: all_var not defined

  1. no_solve_sub() @ Other: 18
  2. top-level scope @ Local: 1  [inlined]
```

```
T = no_solve_sub()[1]
```

```
UndefVarError: all_var not defined

  1. no_solve_sub() @ Other: 18
  2. top-level scope @ Local: 1  [inlined]
```

```
h = no_solve_sub()[2]
```

UndefVarError: all_var not defined

1. **no_solve_sub()** @ ⎡ *Other: 18*
2. **top-level scope** @ ⎡ *Local: 1* [inlined]

- `no_solve_sub()[3]`