DAVID BAKER

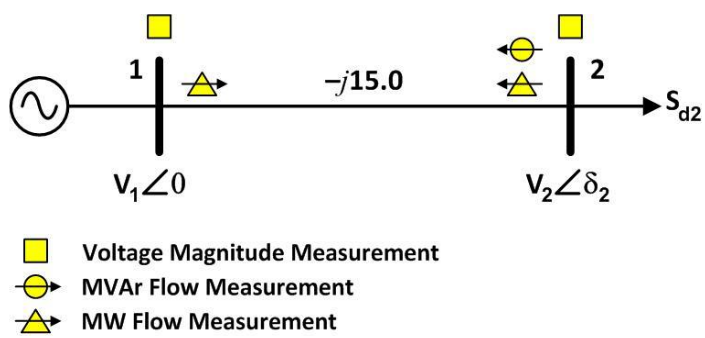
EENG 581: Power System Operation and Management

HW2: State Estimation

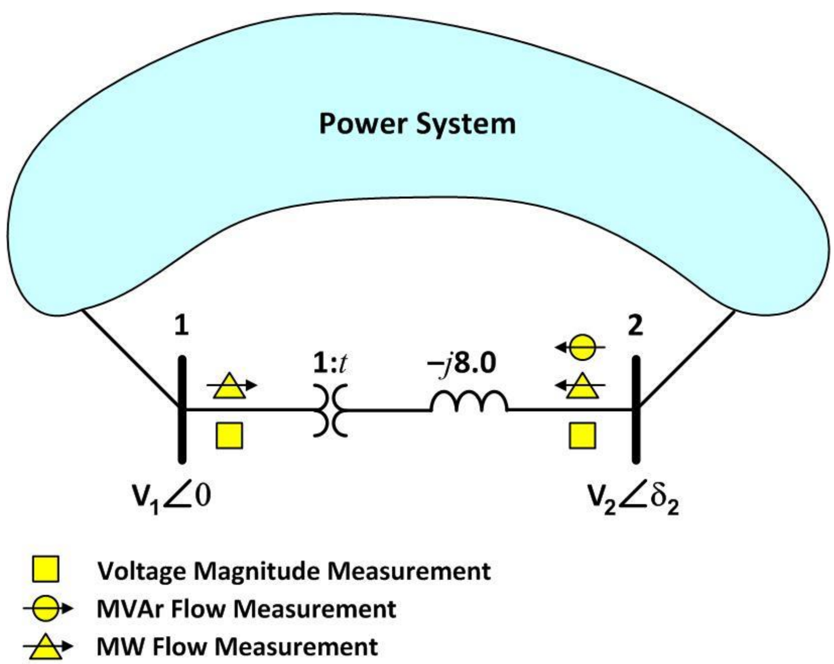
Learning Objectives:

- Formulate the state estimation problem for a power transmission system using the weighted least squares (WLS) method.
- Write a code to iteratively solve the state estimation problem.

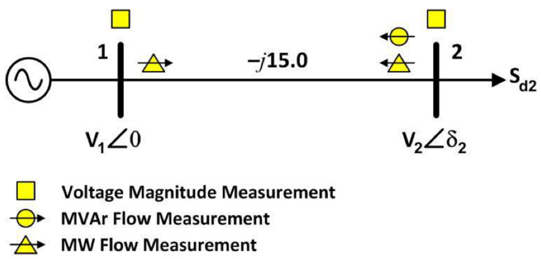
1. Consider the simplified 2-bus power system with the indicated measurements. At a specific instance of time, the following measurements are taken: $V_1 = 1.0$, $V_2 = 0.98$, $P_{12} = 1.65$, $P_{21} = -1.62$, and $Q_{21} = -0.23$ (all values are in per-unit). The standard deviation of the error in all measurements is 4.5%, except for that of V_2 which is zero, i.e., it is measured with absolute accuracy. What does that mean for our state estimator? Form the vector of system states and compute two iterations of the WLS state estimation.



2. Consider an off-nominal tap transformer in a power system. In order to estimate the value of the tap ratio t, 5 measurements are taken as $V_1 = 0.933$, $V_2 = 0.917$, $P_{12} = 0.245$, $P_{21} = -0.240$, and $Q_{21} = 0.170$. The standard deviation of error in the voltage measurements is zero but is 2.5% for the powers. What does this mean for our state estimator? Include the tap setting t in the set of system states and formulate the WLS state estimation algorithm. Perform two iterations of the algorithm.



1. Consider the simplified 2-bus power system with the indicated measurements. At a specific instance of time, the following measurements are taken: $V_1 = 1.0$, $V_2 = 0.98$, $P_{12} = 1.65$, $P_{21} = -1.62$, and $Q_{21} = -0.23$ (all values are in per-unit). The standard deviation of the error in all measurements is 4.5%, except for that of V_2 which is zero, i.e., it is measured with absolute accuracy. What does that mean for our state estimator? Form the vector of system states and compute two iterations of the WLS state estimation.



The fact that σ of V_2^m is zero means that V_2 will be exactly V_2^m, and we can treat V_2 as a fixed value, so we can remove it from our state estimator as a variable.

$$B = \begin{bmatrix} -15 & 15 \\ 15 & -15 \end{bmatrix}$$

$$x = \begin{bmatrix} \delta_2 \\ V_1 \\ V_2 \end{bmatrix} \qquad (\delta_1 = 0)$$

but since V_2 is known exactly, we can just solve for

$$x = \begin{bmatrix} \delta_2 \\ V_1 \end{bmatrix}$$

$n = 2$

unnecessary because V_2 is fixed

$$z = h(x)$$

$$\begin{bmatrix} V_1^m \\ V_2^m \\ P_{12}^m \\ P_{21}^m \\ Q_{21} \end{bmatrix} = \begin{bmatrix} V_1 \\ V_2 \\ V_1 V_2 B_{12} \sin\delta_{12} \\ V_2 V_1 B_{21} \sin\delta_{21} \\ -V_2 V_1 B_{21} \cos\delta_{21} + V_2^2 B_{21} \end{bmatrix}$$

$m = 4$

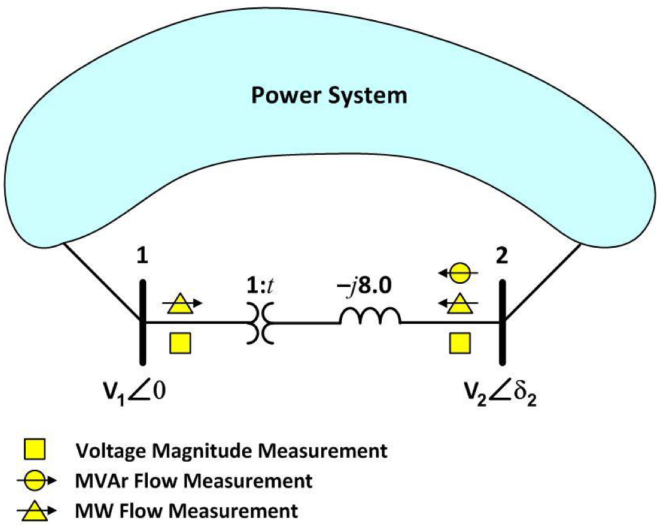
$$H = \frac{\partial h}{\partial x} = \begin{bmatrix} \frac{\partial h_1}{\partial x_1} & \frac{\partial h_1}{\partial x_2} \\ \frac{\partial h_2}{\partial x_1} & \vdots \\ \vdots & \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ -V_1 V_2 B_{12} \cos\delta_{12} & V_2 B_{12} \sin\delta_{12} \\ V_2 V_1 B_{21} \cos\delta_{21} & V_2 B_{21} \sin\delta_{21} \\ V_2 V_1 B_{21} \sin\delta_{21} & -V_2 B_{21} \cos\delta_{21} \end{bmatrix}$$

iteration	δ_2 radians	V_1 pu
flat start	0.0	1.0
1	-0.111224	0.995666
2	-0.11125	1.00183

Code on following page

```julia
using LinearAlgebra
using DataFrames
using PrettyTables

# make fun B subscripts, only works for single digit indices
function symbolFn(i, j)
    subscript = String([Char(0x2080 + i), Char(0x2080 + j)])
    return Symbol("B", subscript)
end

function make_fun_B_subscripts(B)
    for row ∈ 1:length(eachrow(B))
        for col ∈ 1:length(eachcol(B))
            eval(:($(symbolFn(row, col)) = $(B[row, col])))
        end
    end
end

# constants
v₂ = 0.98
δ₁ = 0


function h(x)
    δ₂ = x[1]
    v₁ = x[2]

    return [
        v₁
        v₁ * v₂ * B₁₂ * sin(δ₁ - δ₂)
        v₂ * v₁ * B₂₁ * sin(δ₂ - δ₁)
        -v₂ * v₁ * B₂₁ * cos(δ₂ - δ₁) + v₂^2 * B₂₁
    ]

    return z
end

# susceptance matrix
B = [-15 15
    15 -15]

make_fun_B_subscripts(B)

σ = 0.045 # 4.5% standard deviation of error
W = Diagonal(ones(4) .* (σ^2)) # I think it is sigma squared but not sure

# measurements
Vᵐ₁ = 1.0
Pᵐ₁₂ = 1.65
Pᵐ₂₁ = -1.62
Qᵐ₂₁ = -0.23
z = [Vᵐ₁, Pᵐ₁₂, Pᵐ₂₁, Qᵐ₂₁]

# x is [δ₂, v₁]
# array of x's
arr_x = [[0.0, 1.0]]

for i in 1:2
    x = arr_x[i]
    δ₂ = x[1]
    v₁ = x[2]

    H = [0 1;
        -v₁*v₂*B₁₂*cos(δ₁ - δ₂) v₂*B₁₂*sin(δ₁ - δ₂)
        v₂*v₁*B₂₁*cos(δ₂ - δ₁) v₂*B₂₁*sin(δ₂ - δ₁)
        v₂*v₁*B₂₁*sin(δ₂ - δ₁) -v₂*B₂₁*cos(δ₂ - δ₁)
    ]


    x_new = x + ((H' * W * H)^-1) * (H' * W * (z - h(x)))
    push!(arr_x, x_new)
end

# display results
df = DataFrame(Iteration=["flat start", 1, 2], δ₂=[xx[1] for xx ∈ arr_x], v₁=[xx[2] for xx ∈ arr_x])
header = (["iteration", "δ₂", "v₁"], ["", "radians", "pu"])
pretty_table(df, header=header, header_crayon=crayon"yellow bold")
```

2. Consider an off-nominal tap transformer in a power system. In order to estimate the value of the tap ratio t, 5 measurements are taken as $V_1 = 0.933$, $V_2 = 0.917$, $P_{12} = 0.245$, $P_{21} = -0.240$, and $Q_{21} = 0.170$. The standard deviation of error in the voltage measurements is zero but is 2.5% for the powers. What does this mean for our state estimator? Include the tap setting t in the set of system states and formulate the WLS state estimation algorithm. Perform two iterations of the algorithm.



Voltages are exact, so they are fixed,

$V_1 = 0.933$

$V_2 = 0.917$,

assign $\delta_1 = 0$

so $x = \begin{bmatrix} \delta_2 \\ t \end{bmatrix}$

We don't really know the full B matrix because we don't know anything about the rest of the system, but we do know $B_{21} = B_{12} = 8$

Replace V_1 with tV_1

$$z = h(x)$$

$$\begin{bmatrix} P_{12} \\ P_{21} \\ Q_{21} \end{bmatrix} = \begin{bmatrix} tV_1 V_2 B_{12} \sin\delta_{12} \\ tV_2 V_1 B_{21} \sin\delta_{21} \\ -tV_2 V_1 B_{21}\cos\delta_{21} + V_2^2 B_{21} \end{bmatrix}$$

$$H = \frac{\partial h}{\partial x} = \begin{bmatrix} -tV_1V_2B_{12}\cos\delta_{12} & V_1V_2B_{12}\sin\delta_{12} \\ tV_2V_1B_{21}\cos\delta_{21} & V_2V_1B_{21}\sin\delta_{21} \\ tV_2V_1B_{21}\sin\delta_{21} & -V_2V_1B_{21}\cos\delta_{21} \end{bmatrix}$$

iteration	δ_2 radians	t tap setting
flat start	0.0	1.0
1	-0.03543	0.958014
2	-0.0369669	0.958667

Code to follow

```julia
using LinearAlgebra
using DataFrames
using PrettyTables

# constants
v_1 = 0.933
v_2 = 0.917
δ_1 = 0

function h(x)
    δ_2 = x[1]
    t = x[2]

    return [
        t * v_1 * v_2 * B_12 * sin(δ_1 - δ_2)
        t * v_2 * v_1 * B_21 * sin(δ_2 - δ_1)
        -t * v_2 * v_1 * B_21 * cos(δ_2 - δ_1) + v_2^2 * B_21
    ]

    return z
end

# susceptance matrix entries
B_12 = B_21 = 8


σ = 0.025# 2.5% standard deviation of error
W = Diagonal(ones(3) .* (σ^2))

# measurements
Pᵐ₁₂ = 0.245
Pᵐ₂₁ = -0.240
Qᵐ₂₁ = 0.170
z = [Pᵐ₁₂, Pᵐ₂₁, Qᵐ₂₁]

# x is [δ_2, t] (where t is tap ratio)
# array of x's
arr_x = [[0.0, 1]]

num_iterations = 2
for i in 1:num_iterations
    x = arr_x[i]
    δ_2 = x[1]
    t = x[2]

    H = [-t*v_1*v_2*B_12*cos(δ_1 - δ_2) v_1*v_2*B_12*sin(δ_1 - δ_2)
        t*v_2*v_1*B_21*cos(δ_2 - δ_1) v_2*v_1*B_21*sin(δ_2 - δ_1)
        t*v_2*v_1*B_21*sin(δ_2 - δ_1) -v_2*v_1*B_21*cos(δ_2 - δ_1)
    ]


    x_new = x + ((H' * W * H)^-1) * (H' * W * (z - h(x)))
    push!(arr_x, x_new)
end

# display results
df = DataFrame(Iteration=["flat start", (1:num_iterations)...], δ₂=[xx[1] for xx ∈ arr_x], t=[xx[2] for xx ∈ arr_x])
header = (["iteration", "δ₂", "t"], ["", "radians", "tap setting"])
pretty_table(df, header=header, header_crayon=crayon"yellow bold")
```