Assignment 2

by Thomas Campbell

Problem 1a

How many types of measurements are included in the observation file for each of the GPS, GLONASS, GALIEO and BeiDou systems?

Answer

From file:

G 18 C1C L1C D1C S1C C1W S1W C2W L2W D2W S2W C2L L2L D2L S2L C5Q L5Q D5Q S5Q

E 20 C1C L1C D1C S1C C6C L6C D6C S6C C5Q L5Q D5Q S5Q C7Q L7Q D7Q S7Q C8Q L8Q D8Q S8Q

R 20 C1C L1C D1C S1C C1P L1P D1P S1P C2P L2P D2P S2P C2C L2C D2C S2C C3Q L3Q D3Q S3Q

C 12 C2I L2I D2I S2I C7I L7I D7I S7I C6I L6I D6I S6I

G => GPS: 18 measurements.
E => Galileo: 20 measurements.
R => GLONASS: 20 measurements.
C => BeiDou: 12 measurements.

1b.)

What are they?

Answer

GPS

L1: {Pseudorange (PR) C/A, Phase C/A, C/A Doppler, C/A Signal Strength, Z-tracking PR, Z-tracking Signal Strength},

L2: {Z-tracking PR, Z-tracking Phase, Z-tracking Doppler, Z-tracking Signal Strength, L-code PR, L-code Phase, L-code Doppler, L-code Signal Strength},

L5: {Q PR, Q Phase, Q Doppler, Q Signal Strength}

Galileo

E1: {C PR, C Phase, C Doppler, C Signal Strength},

E6: {C PR, C Phase, C Doppler, C Signal Strength},

E5: {Q PR, Q Phase, Q Doppler, Q Signal Strength},

E7: {Q PR, Q Phase, Q Doppler, Q Signal Strength},

E8: {Q PR, Q Phase, Q Doppler, Q Signal Strength}

GLONASS

G1: {C/A PR, C/A Phase, C/A Doppler, C/A Signal Strength, P PR, P Phase, P Doppler, P Signal Strength},

G2: {P PR, P Phase, P Doppler, P Signal Strength, C/A PR, C/A Phase, C/A Doppler, C/A Signal Strength},

G3: {Q PR, Q Phase, Q Doppler, Q Signal Strength}

BeiDou

B1-2: {I PR, I Phase, I Doppler, I Signal Strength},

B2b: {I PR, I Phase, I Doppler, I Signal Strength},

B3: {I PR, I Phase, I Doppler, I Signal Strength}

1c.)

Select one satellite from each of the systems (e.g., GPS01, Galileo02, GLONASS15), and plot the corresponding pseudorange measurements (you can choose any code type) as a function of time on a same figure. Make sure the all elements in the figure are clearly labeled

In [ ]:

```
using DataFrames, Dates, Plots, Statistics, LaTeXStrings
default(seriestype=:scatter, markerstrokewidth=0, xrotation=45, markersize=0.5, right_margin=1
c = 299792458
F_0 = 10.23
F_1 = 154
F_2 = 120
F_5 = 115

F_12 = 152.6
F_2b = 118
F_3 = 124

F_1r = (1602 - 9 / 16) / 10.23
F_2r = (1246 - 7 / 16) / 10.23

λ_1_g = c / (F_0 * F_1 * 1e6)
λ_2_g = c / (F_0 * F_2 * 1e6)
λ_5_g = c / (F_0 * F_5 * 1e6)

λ_1_r = c / ((1602 - 9 / 16) * 1e6)
λ_2_r = c / ((1246 - 7 / 16) * 1e6)

λ_12_c = c / (F_0 * F_12 * 1e6)
λ_2b_c = c / (F_0 * F_2b * 1e6)
λ_3_c = c / (F_0 * F_3 * 1e6)

function try_parse(str)
    try
        parse(Float64, str)
    catch e
        return missing
    end
end

function parse_rinex(in_path)

    # read in sp3 file
    in_stream = open(in_path, "r")
    in_lines = readlines(in_stream)

    # the first 22 lines are not needed for these purposes.
    data = in_lines[55:length(in_lines)-1]

    # initialize vectors for each kind of data. sp_d is an intermediate vector, used only to u
    dates = []
    entries = []
    sp_d = []
    obs = []

    # iterate over the sp3 dataset. at each row, split, then push to either dates, positions, 
    # at each occurrence of a position data line, sp_d will also be pushed to dates.
    # this way, if the number of PRN's changes over the course of the day, the number of repea
    for line in eachindex(data)
        ln = data[line]

        ln = rpad(ln, 321, " ")

        sp = split(ln)
        if sp[1] == ">"
            sp_d = copy(sp)

        else
```

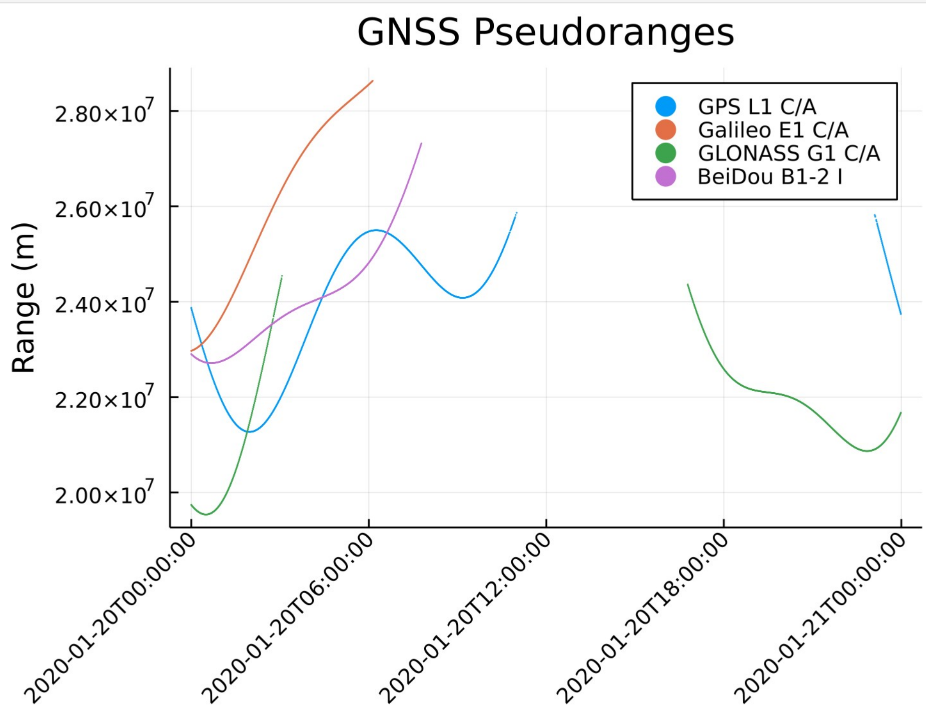
```
            push!(entries, try_parse.([ln[4:17], ln[20:33], ln[36:49], ln[52:65], ln[68:81], l
                ln[116:129], ln[132:145], ln[148:161], ln[164:177], ln[180:193], ln[196:209],
                ln[212:225], ln[228:241], ln[244:257], ln[260:273], ln[276:289],
                ln[292:305], ln[308:321]]))

            push!(dates, parse.(Float64, sp_d[2:end-2]))
            push!(obs, ln[1:3])
        end
    end

    # create a matrix of parsed sp3 data
    return vcat.(dates, obs, entries)

end

function make_rindf(rinex, gnss)

    col_names = Dict("gps" => ["C1C", "L1C", "D1C", "S1C", "C1W", "S1W", "C2W", "L2W", "D2W",
        "galileo" => ["C1C", "L1C", "D1C", "S1C", "C6C", "L6C", "D6C", "S6C", "C5Q", "L5Q", "D
        "sbas" => ["C1C", "L1C", "D1C", "S1C", "C5I", "L5I", "D5I", "S5I"],
        "glonass" => ["C1C", "L1C", "D1C", "S1C", "C1P", "L1P", "D1P", "S1P", "C2P", "L2P", "D
        "bds" => ["C2I", "L2I", "D2I", "S2I", "C7I", "L7I", "D7I", "S7I", "C6I", "L6I", "D6I",
        "qzss" => ["C1C", "L1C", "D1C", "S1C", "C2L", "L2L", "D2L", "S2L", "C5Q", "L5Q", "D5Q"
        "irnss" => ["C5A", "L5A", "D5A", "S5A"])

    if gnss == "gps"
        gnss_flag = 'G'
    elseif gnss == "galileo"
        gnss_flag = 'E'
    elseif gnss == "sbas"
        gnss_flag = 'S'
    elseif gnss == "glonass"
        gnss_flag = 'R'
    elseif gnss == "bds"
        gnss_flag = 'C'
    elseif gnss == "qzss"
        gnss_flag = 'J'
    elseif gnss == "irnss"
        gnss_flag = 'I'
    end

    gnss_data = []

    for line in eachindex(rinex)
        if rinex[line][7][1] == gnss_flag
            push!(gnss_data, rinex[line])
        end
    end

    full_names = ["year", "month", "day", "hour", "minute", "seconds", "sat"]
    date_names = copy(full_names[1:end-1])
    append!(full_names, col_names[gnss])

    df = DataFrame(mapreduce(permutedims, vcat, gnss_data), :auto)
    df = df[:, collect(1:length(full_names))]

    rename!(df, Symbol.(full_names))

    df.dt = DateTime.(Dates.Year.(df.year),
        Dates.Month.(df.month),
        Dates.Day.(df.day),
        Dates.Hour.(df.hour),
        Dates.Minute.(df.minute))
```

```
    select!(df, Not(date_names))
    return df
end
```

make_rindf (generic function with 1 method)

In [ ]:
```
in_path = "C:/Users/tom_j/Downloads/ABMF00GLP_R_20200200000_01D_30S_MO.rnx"

# Parse Rinex file
parsed = parse_rinex(in_path);

# Create Datasets
mygps = make_rindf(parsed, "gps");
mygalileo = make_rindf(parsed, "galileo");
myglonass = make_rindf(parsed, "glonass");
mybeidou = make_rindf(parsed, "bds");

# Pick a satellite
g30 = dropmissing(mygps[mygps.sat .== "G30", :]);
e11 = mygalileo[mygalileo.sat .== "E11", :];
r12 = myglonass[myglonass.sat .== "R12", :];
c35 = mybeidou[mybeidou.sat .== "C35", :];
```

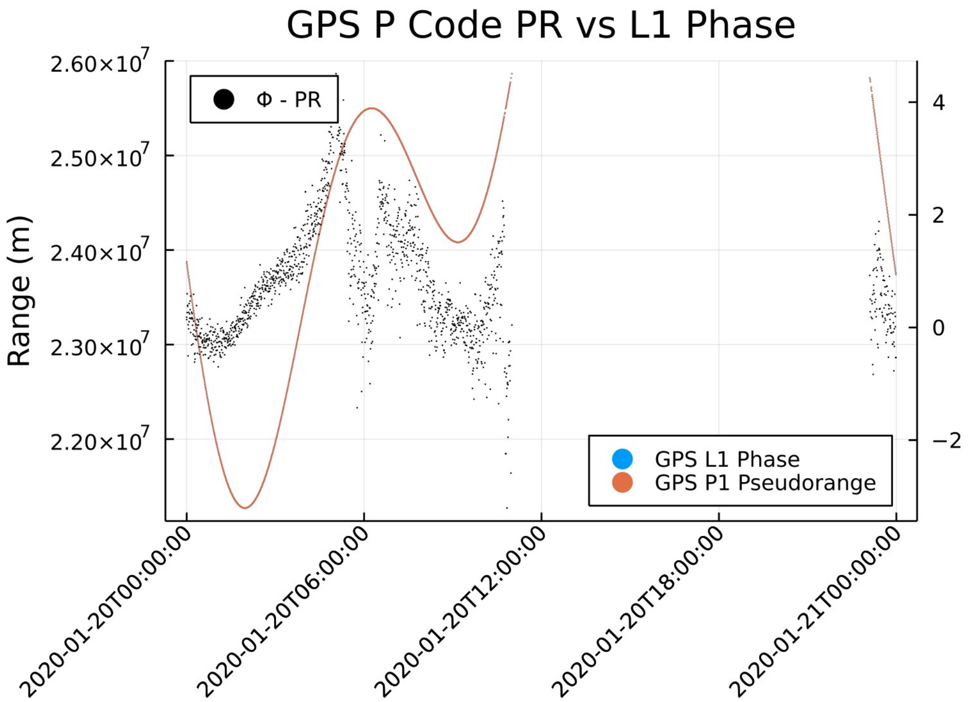
In [ ]:
```
plot(g30.dt, g30.C1C, label= "GPS L1 C/A", title = "GNSS Pseudoranges")
plot!(e11.dt, e11.C1C, label="Galileo E1 C/A")
plot!(r12.dt, r12.C1C,label="GLONASS G1 C/A")
plot!(c35.dt, c35.C2I, label="BeiDou B1-2 I")
```



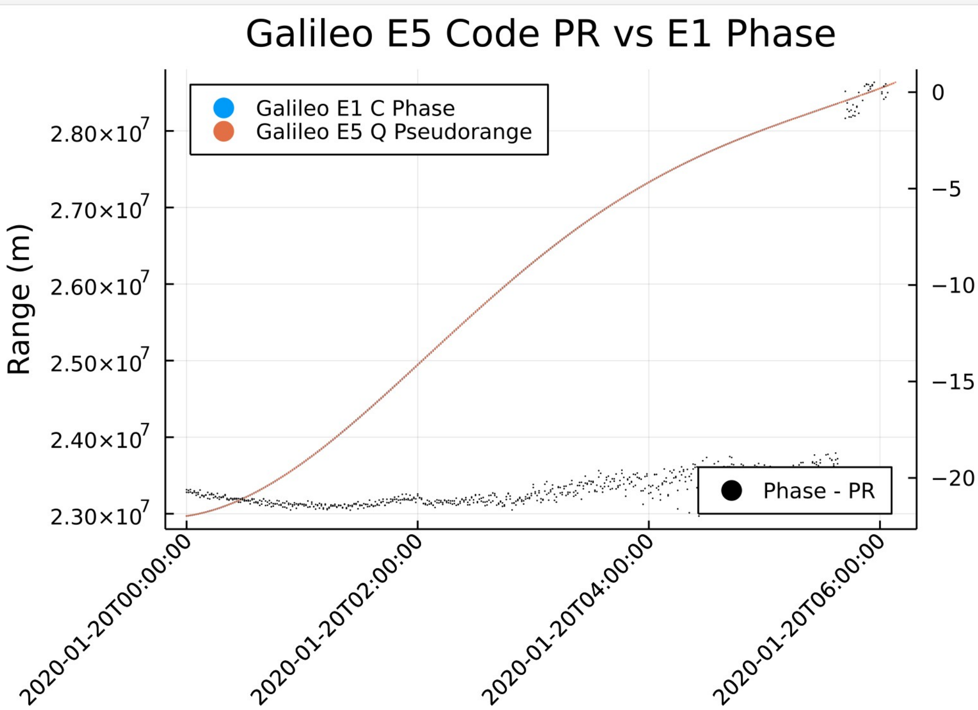
Problem 2a.)

Plot phase L1 and code P1 on a same plot for each of the satellite in problem 1.

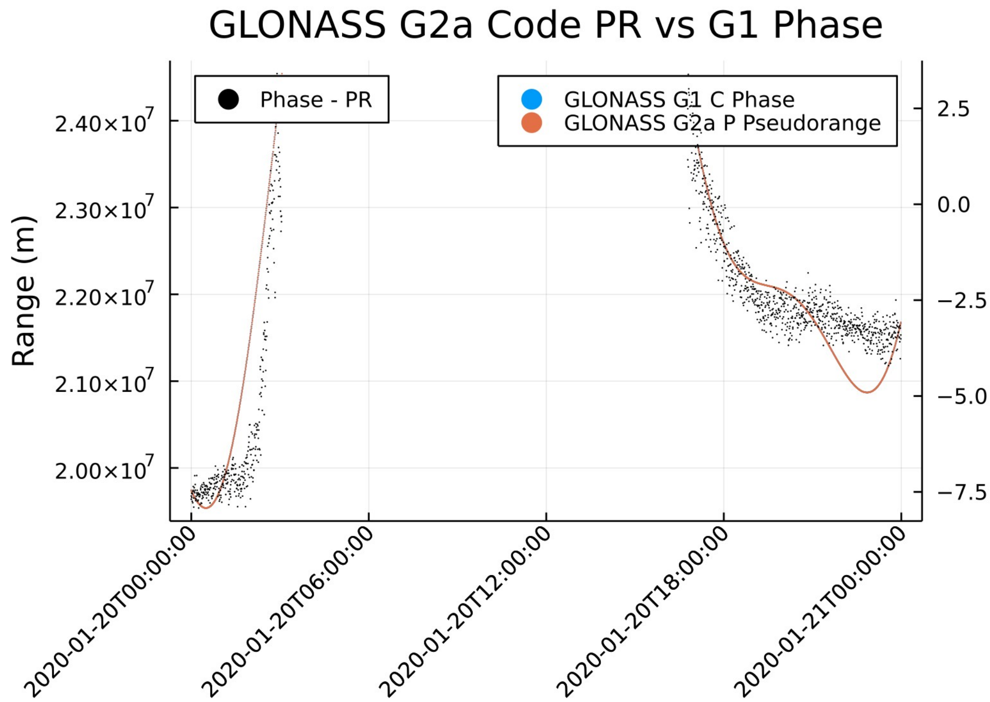
In [ ]:
```
plot(g30.dt, g30.L1C * λ_1, label="GPS L1 Phase", title="GPS P Code PR vs L1 Phase", legend =
plot!(g30.dt, g30.C1C, label="GPS P1 Pseudorange")
plot!(twinx(),g30.dt, g30.C1C - g30.L1C * λ_1 , label= "Φ - PR", legend=:topleft, markercolor
```



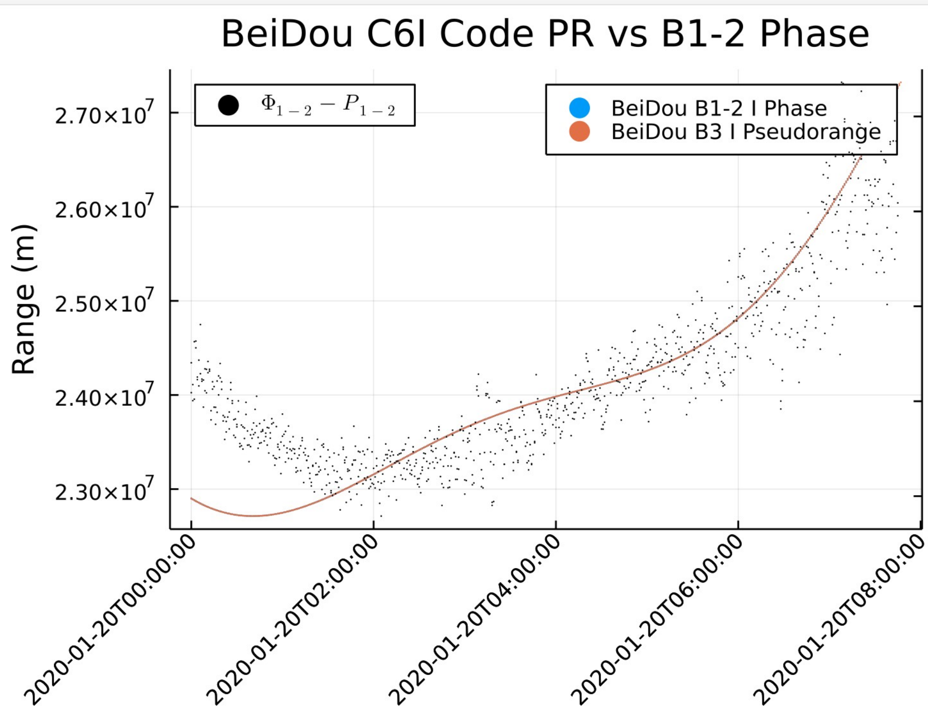
In [ ]:
```
plot(e11.dt, e11.L1C * λ_1,label="Galileo E1 C Phase", title="Galileo E5 Code PR vs E1 Phase",
plot!(e11.dt, e11.C1C,label="Galileo E5 Q Pseudorange")
plot!(twinx(), e11.dt,  e11.C1C - e11.L1C * λ_1, label="Phase - PR",legend=:bottomright, marke
```



In [ ]:
```
plot(r12.dt, r12.L1C * λ_1_r,label="GLONASS G1 C Phase", title="GLONASS G2a Code PR vs G1 Phas
plot!(r12.dt, r12.C1C, label="GLONASS G2a P Pseudorange")
plot!(twinx(), r12.dt, r12.C1C - r12.L1C * λ_1_r , label="Phase - PR", legend=:topleft, marker
```




```
In [ ]: plot(c35.dt, c35.L2I * λ_12_c, label="BeiDou B1-2 I Phase", title="BeiDou C6I Code PR vs B1-2 I
        plot!(c35.dt, c35.C2I, label="BeiDou B3 I Pseudorange")
        plot!(twinx(), c35.dt, c35.C2I - c35.L2I * λ_12_c, label= L"$\Phi_{1-2} - P_{1-2}$", legend=:to
```



2b.)

Would you conclude one is more precise than the other? And why?

Answer

It seems hard to tell which measurement would be more precise from the plots above.

Need to look at the chip rates and precision for carrier phases. Maybe they are close given the different codes?

Problem 3a.)

Why can ionosphere-free combinations remove the first-order ionospheric effect?

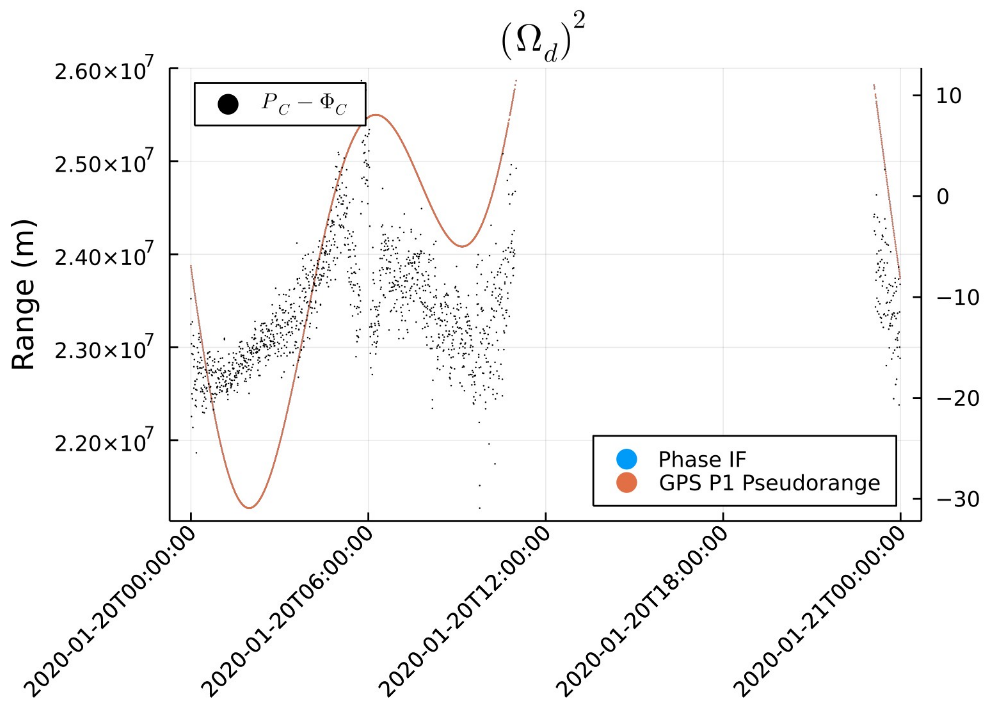
Answer

The ionosphere-free combination remove the first-order ionospheric effect because each measurement is multiplied by its respective frequency twice. This new measurement f_i^2 * R_i, has an ionospheric error that is independent of frequency. Thus, by differencing with another f_j^2 * R_j, the ionospheric errors should cancel out.

3b.)

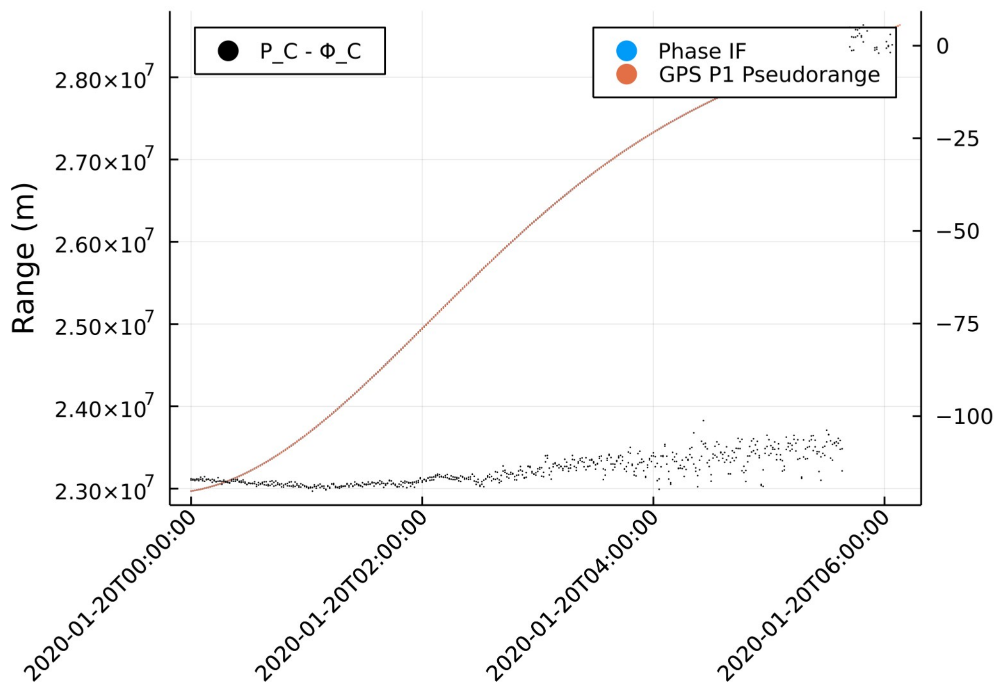
Form code (Pc) and carrier phase (L c) ionosphere-free combinations for the three satellites.

In [ ]:
```
using LaTeXStrings
function ion_free(f_1, f_2, m_1, m_2)
    γ = f_1/f_2
    α = γ*γ/(1 - γ*γ)
    return m_1 + 2*α*(m_1 - m_2)
end

#plot(g30.dt, ion_free(F_1, F_2, g30.C1C, g30.C2W), markerstrokewidth=0, xrotation=45, label="

plot(g30.dt, ion_free(F_1, F_2, λ_1_g * g30.L1C, λ_2_g * g30.L2W), label="Phase IF", legend =
plot!(g30.dt, ion_free(F_1, F_2, g30.C1C, g30.C2W),label="GPS P1 Pseudorange")
plot!(twinx(), g30.dt, ion_free(F_1, F_2, g30.C1C, g30.C2W) - ion_free(F_1, F_2, λ_1_g * g30.L
```



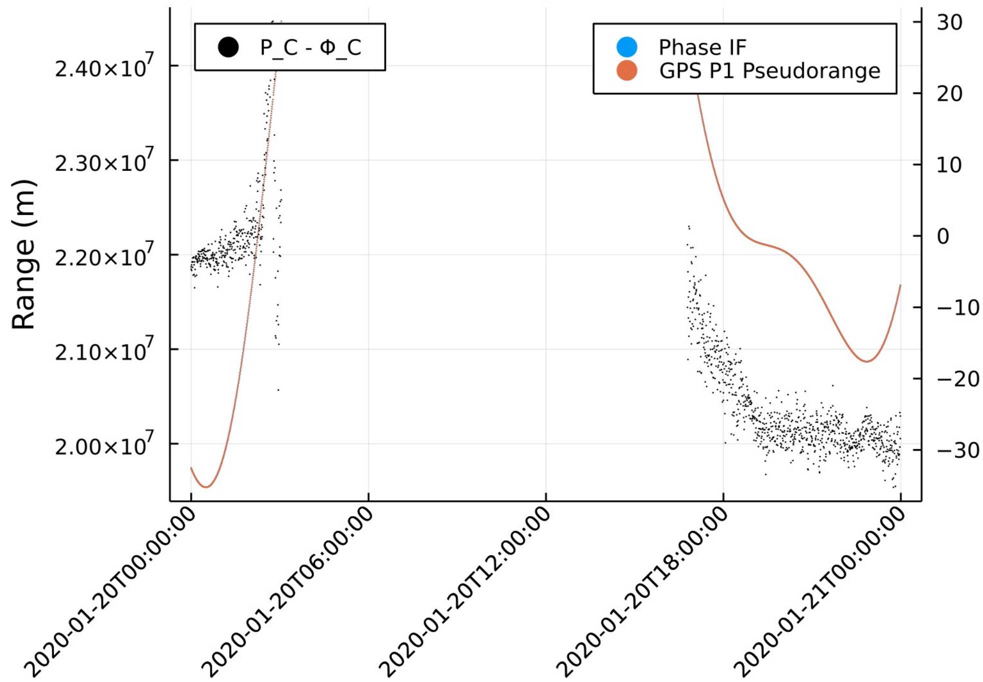
In [ ]:

```
plot(e11.dt, ion_free(F_1, F_5, λ_1_g * e11.L1C, λ_5_g * e11.L5Q), label="Phase IF")
plot!(e11.dt, ion_free(F_1, F_5, e11.C1C, e11.C5Q), label="GPS P1 Pseudorange")
plot!(twinx(), e11.dt, ion_free(F_1, F_5, e11.C1C, e11.C5Q) - ion_free(F_1, F_5, λ_1_g * e11.L
```

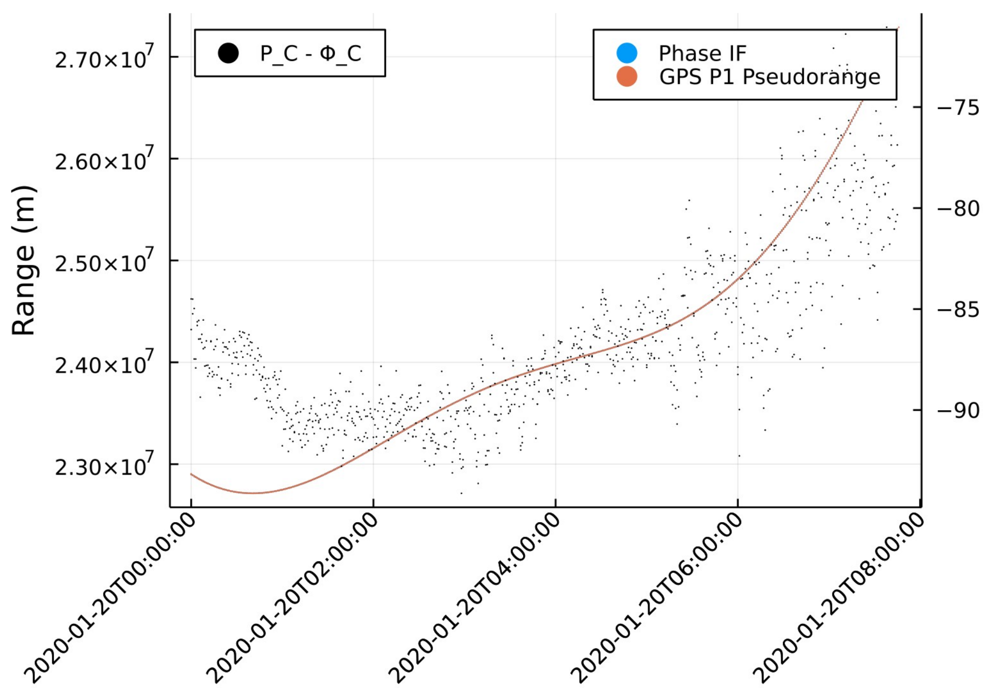


In [ ]:

```
plot(r12.dt, ion_free(F_1r, F_2r, λ_1_r * r12.L1C, λ_2_r * r12.L2P), label="Phase IF")
plot!(r12.dt, ion_free(F_1r, F_2r,r12.C1C, r12.C2P),label="GPS P1 Pseudorange")
plot!(twinx(), r12.dt, ion_free(F_1r, F_2r, r12.C1C, r12.C2P) - ion_free(F_1r, F_2r, λ_1_r * r
```



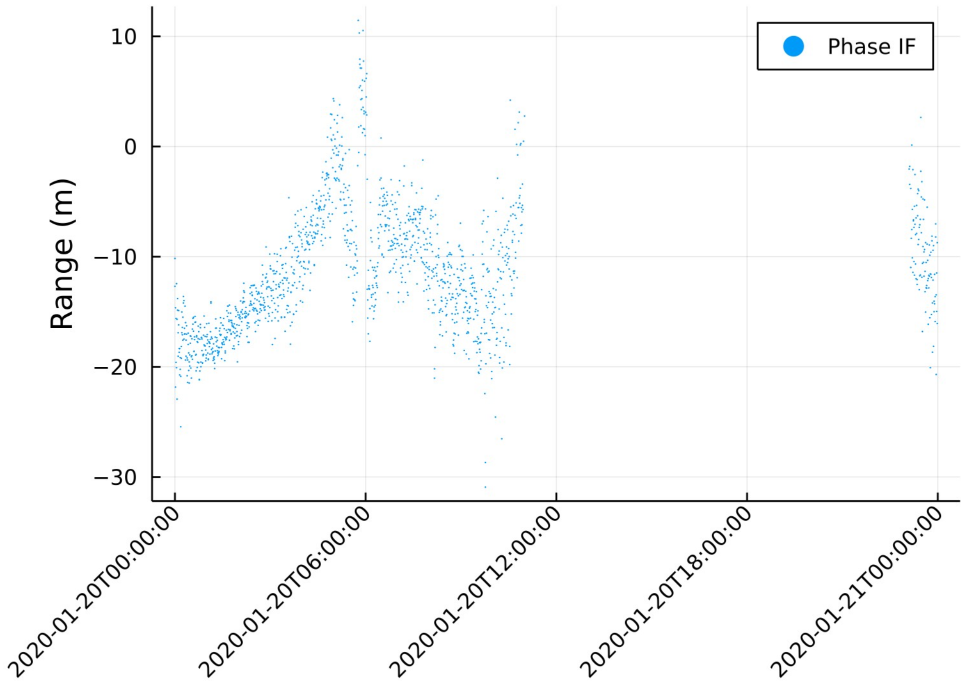

```
In [ ]: plot(c35.dt, ion_free(F_12, F_3, λ_12_c * c35.L2I, λ_3_c * c35.L6I), label="Phase IF")

        plot!(c35.dt, ion_free(F_12, F_3, c35.C2I, c35.C6I), label="GPS P1 Pseudorange")
        plot!(twinx(), c35.dt, ion_free(F_12, F_3, c35.C2I, c35.C6I) - ion_free(F_12, F_3, λ_12_c * c3
```



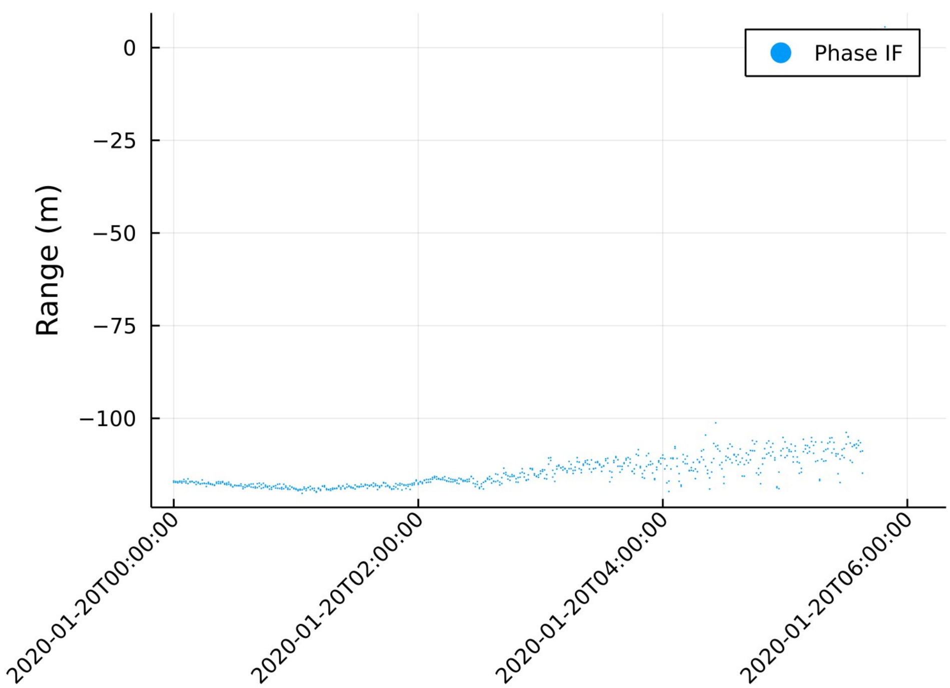
Problem 4a.)

The code multipath can be check by plotting the difference of code and carrier ionosphere-free combinations (Pc - L c). Form the combination for the three satellites.

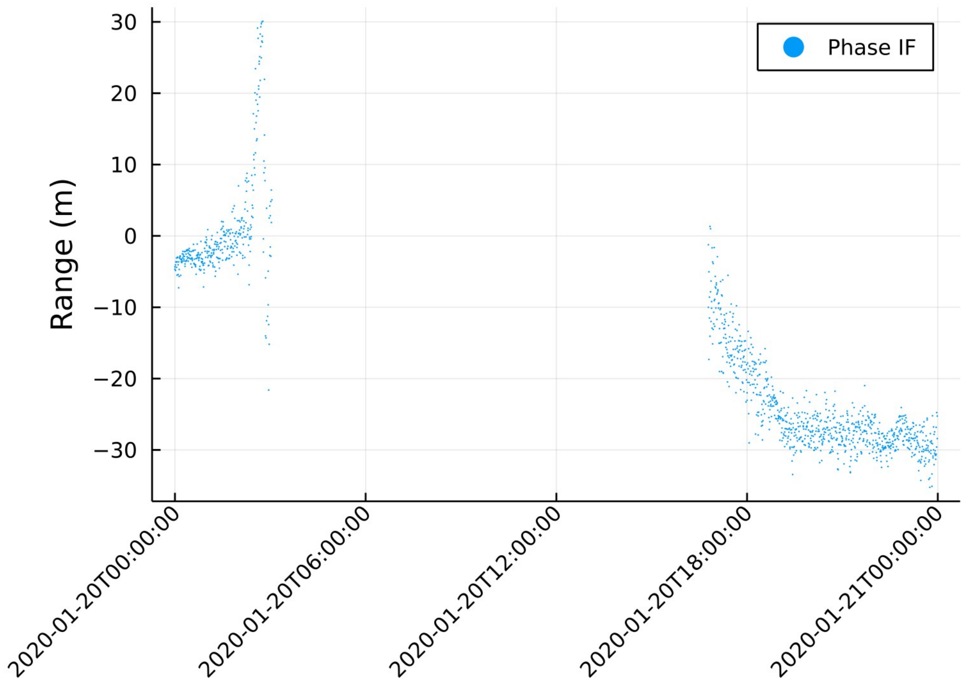
In [ ]:
```
plot(g30.dt, ion_free(F_1, F_2,g30.C1C, g30.C2W) - ion_free(F_1, F_2, λ_1 * g30.L1C, λ_2 * g30
#plot(g30.dt, g30.C1W - ion_free(F_1, F_2, λ_1_g * g30.L1C, λ_2_g * g30.L2W))
```



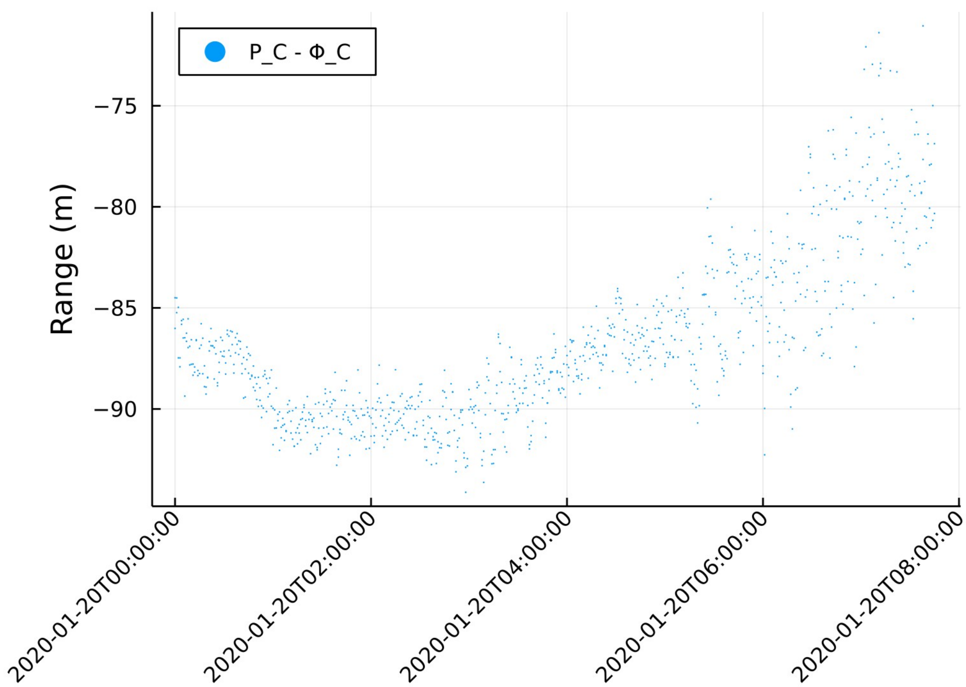
In [ ]:
```
plot(e11.dt, ion_free(F_1, F_5, e11.C1C, e11.C5Q) - ion_free(F_1, F_5, λ_1_g * e11.L1C, λ_5_g
```



In [ ]:
```
plot(r12.dt, ion_free(F_1r, F_2r, r12.C1C, r12.C2P) - ion_free(F_1r, F_2r, λ_1_r * r12.L1C, λ_
```



In [ ]:
```
plot(c35.dt, ion_free(F_12, F_3, c35.C2I, c35.C6I) - ion_free(F_12, F_3, λ_12_c * c35.L2I, λ_3_
```



4b.)

What can you tell from the results?

Problem 5a.)

What is geometry-free combination, and what is the physical meaning of this combination?

Answer

The geometry-free combination is the simple difference between a signal type from one frequency and the same signal type from another frequency (i.e. P_1 - P_2 or Phi_2 - Phi_1). Because the difference is across frequencies, all errors that do not depend on frequency are cancelled out.

5b.)

What are the main utilities of the combination?

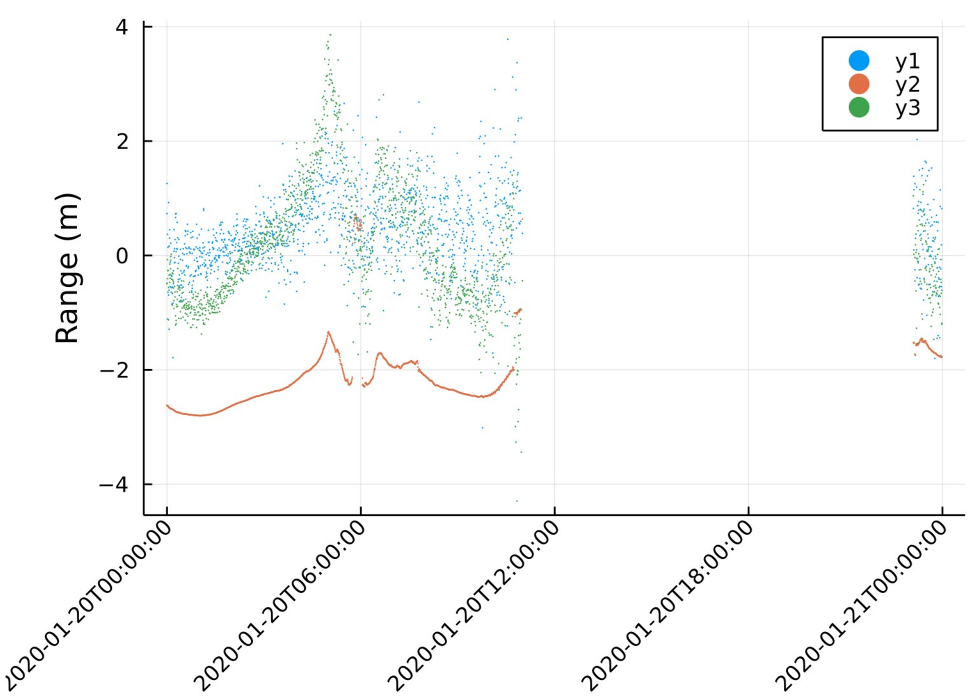
Answer

Because the geometry-free combination removes the range and troposphere, it can be used to estimate the ionosphere and multipath. It can also be used to detect cycle slips when observing carrier phase measurements, since the integer ambiguity's effect is dependent on frequency. This means that after a loss of lock, the integer ambiguity will be different than before, and can show as a up or down shift in the geometry free combination.

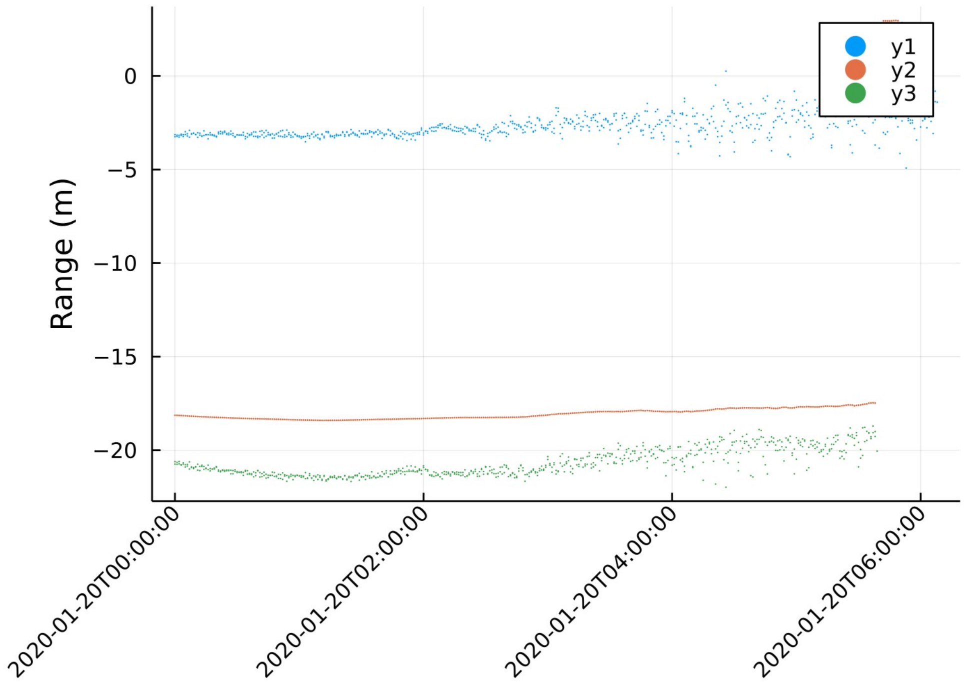
5c.)

Form the geometry-free combination, e.g., L1-L2, P1-P2 and P1-L1, for the three satellites.

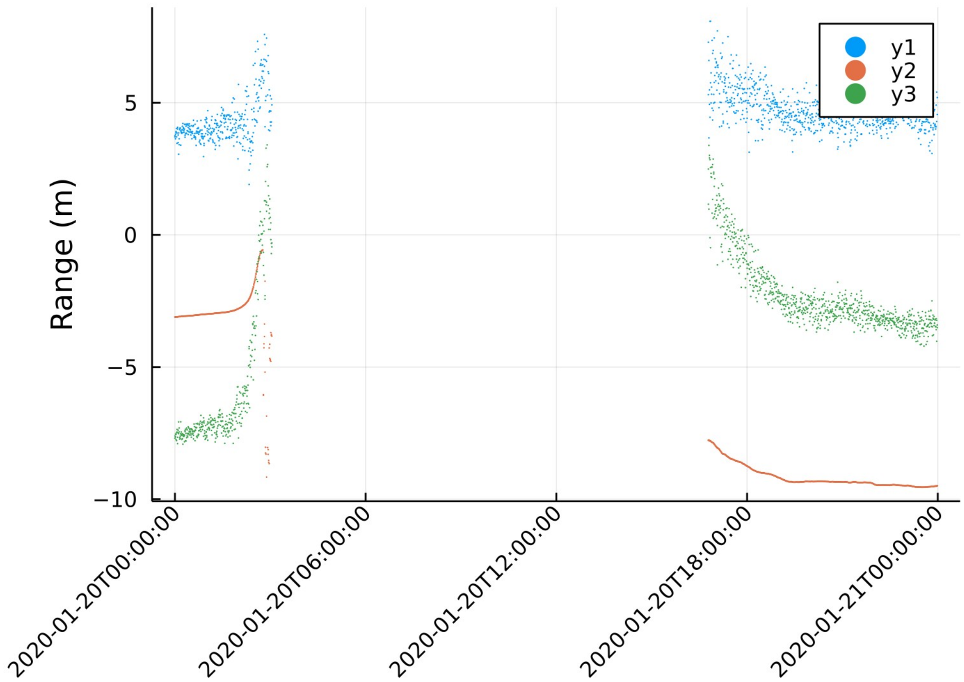
In [ ]:
```
plot(g30.dt, g30.C2W - g30.C1W)

plot!(g30.dt, λ_1_g * g30.L1C - λ_2_g * g30.L2W)

plot!(g30.dt,g30.C1W - λ_1 * g30.L1C )

#Why are these all shifted around?
```



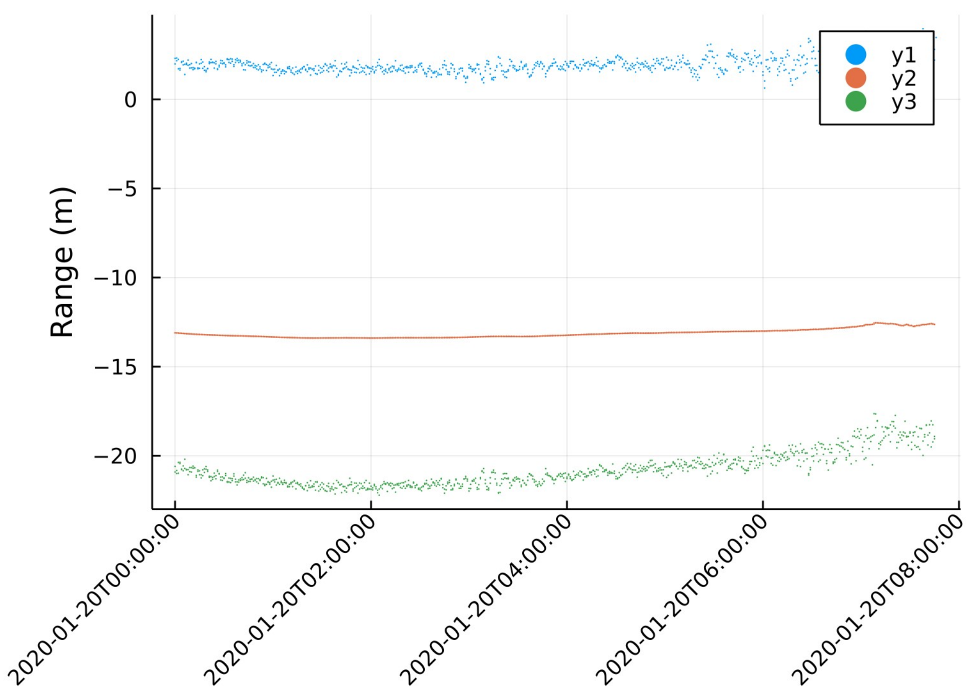
In [ ]:
```
plot(e11.dt, e11.C5Q - e11.C1C)

plot!(e11.dt, λ_1_g * e11.L1C - λ_5_g * e11.L5Q)

plot!(e11.dt, e11.C1C - λ_1_g * e11.L1C)

#plot(e11.dt, ion_free(F_1, F_5, e11.C1C, e11.C5Q) - ion_free(F_1, F_5, λ_1_g * e11.L1C, λ_5_g
```



In [ ]:
```
plot(r12.dt, r12.C2P - r12.C1C)

plot!(r12.dt, λ_1_r * r12.L1C - λ_2_r * r12.L2P)

plot!(r12.dt, r12.C1C - λ_1_r * r12.L1C)

#plot(r12.dt, ion_free(F_1r, F_2r, r12.C1C, r12.C2P) - ion_free(F_1r, F_2r, λ_1_r * r12.L1C, λ_
```



In [ ]:
```
plot(c35.dt, c35.C6I - c35.C2I)

plot!(c35.dt, λ_12_c * c35.L2I - λ_3_c * c35.L6I)

plot!(c35.dt, c35.C2I - λ_12_c * c35.L2I)
```



5d.)

Among three combinations (e.g., L1-L2, P1-P2 and P1-L1), which one shows a greater level of noise, and why?

5e.)

How would you explain the variations in these combinations with time?

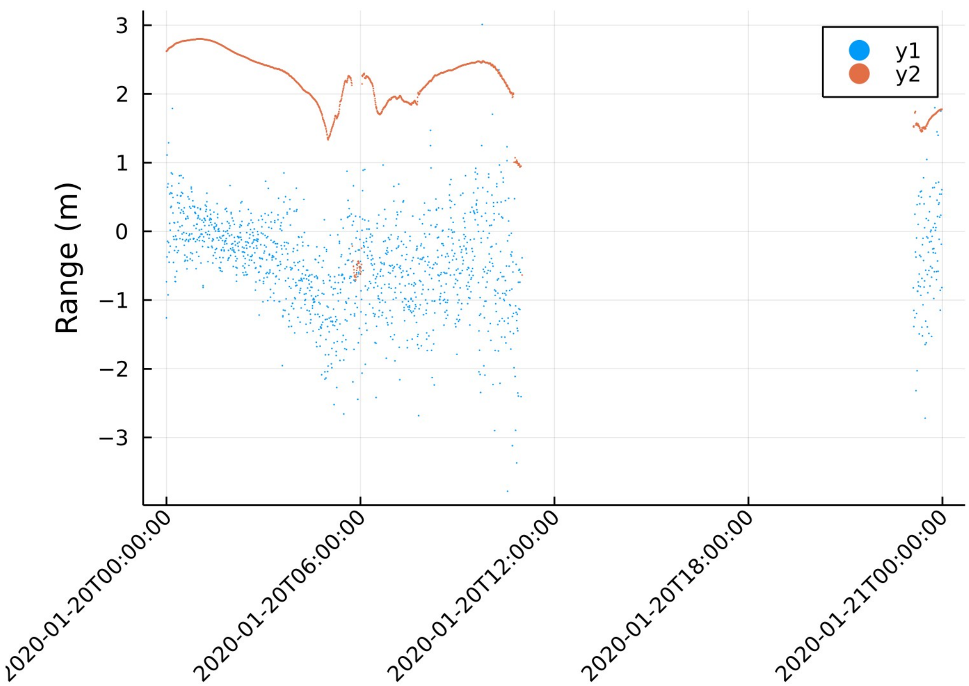
5f.)

Why do the combinations of L1-L2 and P1-P2 have opposite sign?

5g.)

Plot the combination (L1-L2) and (P2-P1) on the same plot.

In [ ]:
```
plot(g30.dt, g30.C1W - g30.C2W)
plot!(g30.dt,λ_2 * g30.L2W - λ_1 * g30.L1C )
```



5h.)

Does dispersion increase at the beginning and end of arcs?

Answer

Yes.

5i.)

How would you explain the increase if you do see the increase?

Arcs usually begin and end at low elevations. At low elevations, signals are more likely to experience obstructions, increased tropospheric and ionospheric delays, as well as more multipath.

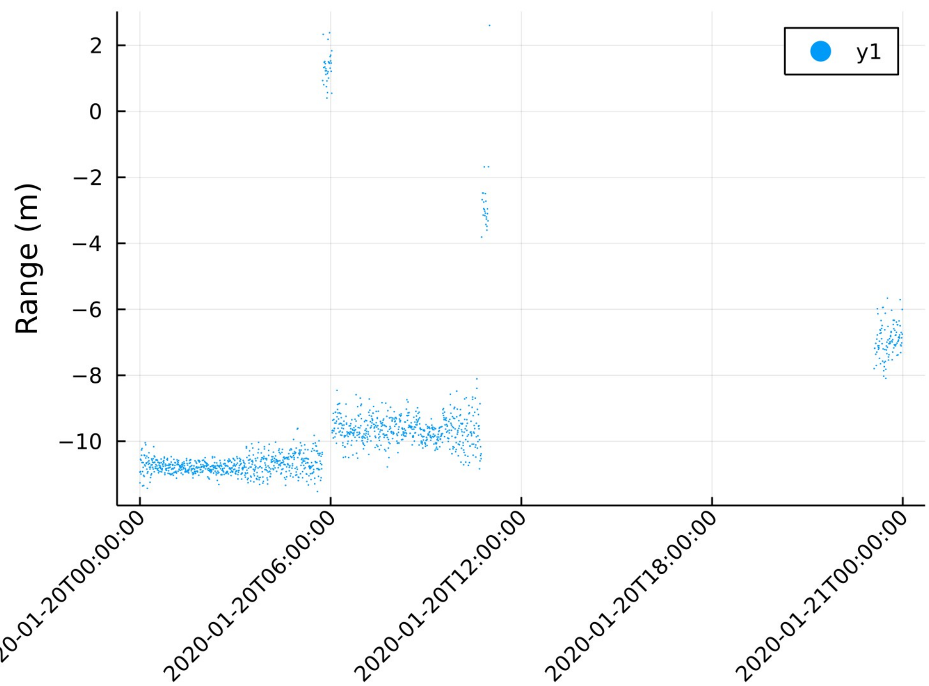
Problem 6a.)

What is MW combination, and what is it used for?

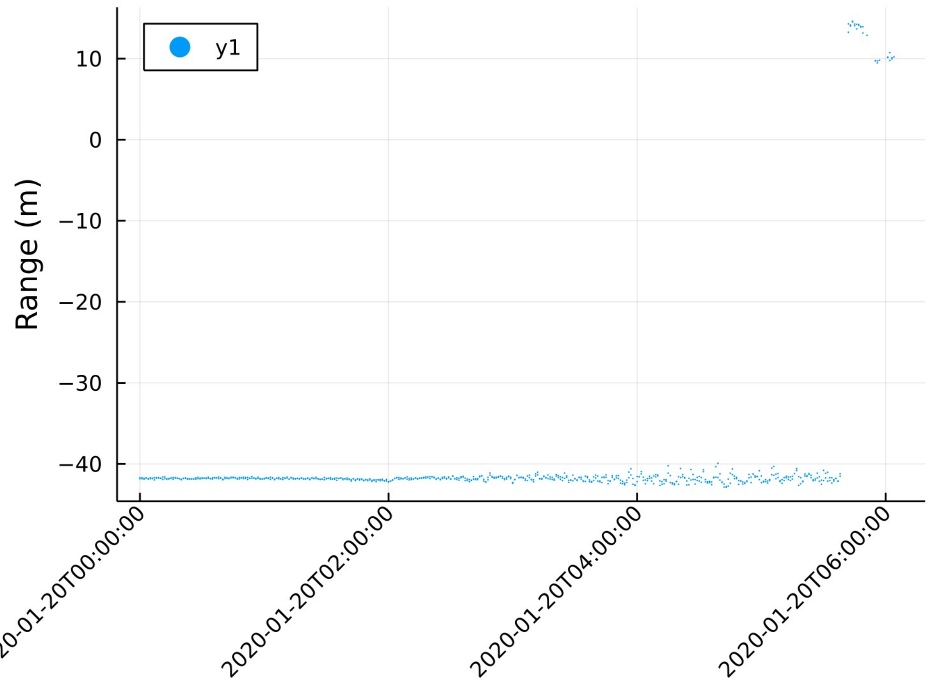
6b.)

Form the Mw combination for the three satellites.

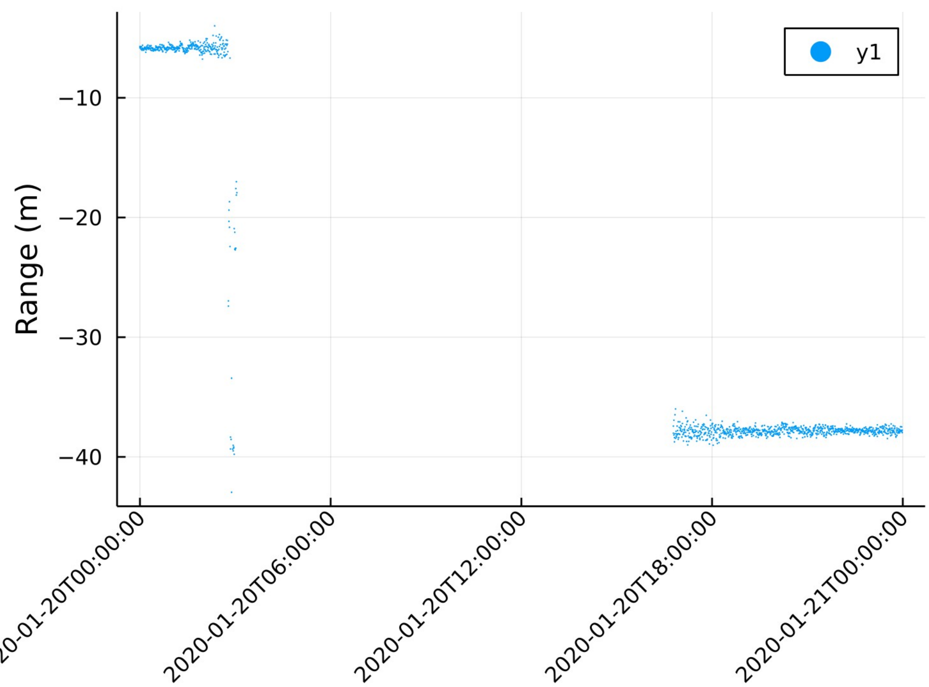
In [ ]:
```
λ_W_inv(λ_x, λ_y) = 1/λ_x - 1/λ_y
narrowlane(f_1, f_2, m_1, m_2) = (f_1*m_1 + f_2*m_2) / (f_1 + f_2)


plot(g30.dt, ( (g30.L1C - g30.L2W) / λ_W_inv(λ_1_g, λ_2_g) - narrowlane(F_1, F_2, g30.C1C, g30
```



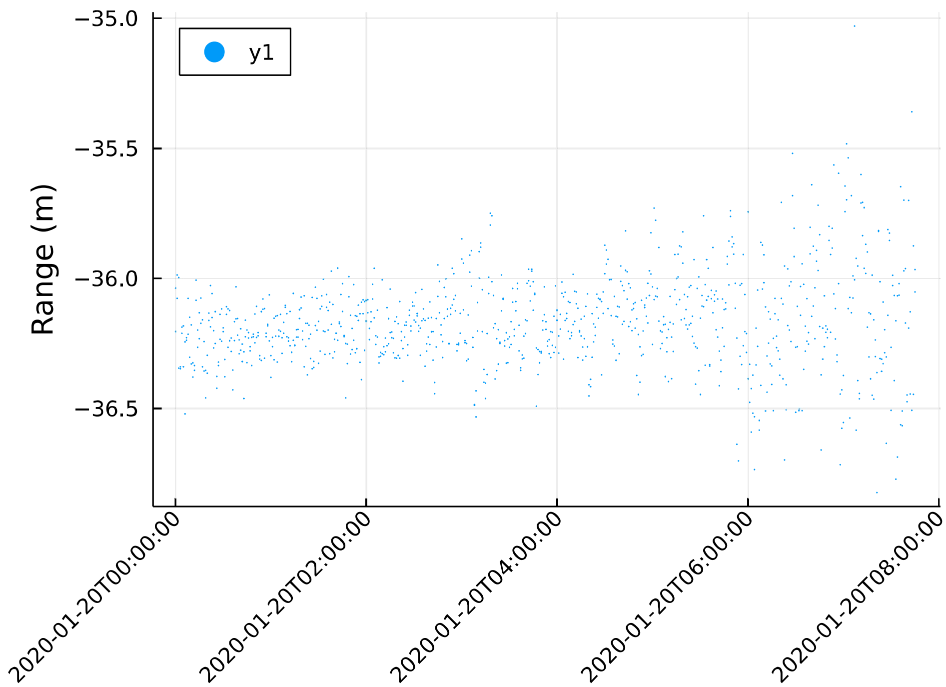
In [ ]:
```
plot(e11.dt, (((e11.L1C - e11.L5Q) / λ_W_inv(λ_1_g, λ_5_g)) - narrowlane(F_1, F_2, e11.C1C, e1
```




```
In [ ]: plot(r12.dt, ((r12.L1C - r12.L2P) / λ_W_inv(λ_1_r, λ_2_r) - narrowlane(F_1, F_2, r12.C1C, r12.(
```




```
In [ ]: plot(c35.dt, ((c35.L2I - c35.L6I) / λ_W_inv(λ_12_c, λ_3_c) - narrowlane(F_12, F_3, c35.C2I, c3!
```



y1
Range (m)
−35.0
−35.5
−36.0
−36.5
2020-01-20T00:00:00
2020-01-20T02:00:00
2020-01-20T04:00:00
2020-01-20T06:00:00
2020-01-20T08:00:00