Intro to Julia
Continuum LA Pre-Course

Yuzhan Jiang

Intro to Julia

By Raj Rao Nadakuditi

An introduction to the Julia programming language. Introduces students to variables, arrays, functions, and everything else that they need to succeed!

6. Operations on arrays

6.1. `mapslices`: function on slices of an array

6.2. Broadcasting with `.`

Complete the code below to compute the submatrix `df_white` which contains data for all the white wines.

Hint: there are multiple ways to do this. You can explicitly take all the white wines, or compute the set of all wines that are not red...

```
df_white = df[df.Color .== "White" ,:]
```

✓ 2s

493 rows × 6 columns

	fixedacidity	volatileacidity	freesulfurdioxide	density	alcohol	Color
	Float64	Float64	Float64	Float64	Float64	String
1	6.6	0.425	23.0	0.99082	11.4	White
2	7.2	0.25	51.0	0.9964	9.2	White
3	6.9	0.25	28.0	0.99088	11.7	White
4	7.7	0.275	19.0	0.992	10.7	White
5	7.1	0.26	31.0	0.99644	11.2	White
6	6.0	0.24	34.0	0.9946	10.4	White
7	6.9	0.25	36.0	0.9948	10.7	White
8	7.5	0.17	65.0	0.997	10.0	White
9	5.1	0.26	26.0	0.99449	9.2	White
10	7.0	0.14	10.0	0.99352	9.9	White
11	6.4	0.31	12.0	0.9919	10.4	White
12	7.1	0.32	52.0	0.998	8.8	White
13	6.5	0.25	29.0	0.99776	10.1	White
14	5.8	0.12	35.0	0.9908	11.4	White
15	7.4	0.19	33.0	0.993	9.6	White
16	6.2	0.25	58.0	0.99454	10.4	White
17	5.9	0.27	43.0	0.9941	10.7	White
18	6.1	0.27	65.0	0.9957	9.0	White
19	8.0	0.4	27.0	0.9935	12.2	White
20	6.6	0.2	35.0	0.99396	9.4	White
21	5.1	0.14	15.0	0.9919	9.2	White
22	6.7	0.26	40.0	0.99479	10.4	White

	fixedacidity	volatileacidity	freesulfurdioxide	density	alcohol	Color
	Float64	Float64	Float64	Float64	Float64	String
23	6.4	0.17	33.0	0.99152	10.4	White
24	6.8	0.2	38.0	0.993	9.1	White
25	5.8	0.17	11.0	0.99202	10.4	White
26	7.1	0.27	26.0	0.99335	11.5	White
27	7.5	0.14	50.0	0.9945	9.6	White
28	6.5	0.19	23.0	0.9937	10.0	White
29	7.5	0.26	33.0	1.0011	8.8	White
30	6.2	0.35	33.0	0.99908	8.8	White
⋮	⋮	⋮	⋮	⋮	⋮	⋮

```julia
@assert all(df_white.Color .== "White")
println("White wines: $(size(df_white, 1))")
```

✓ 8ms

```
White wines: 493
```

7. Plotting

7.1. Multiple plots on the same axes

7.2. Multiple plots alongside each other using the `layout` option

8. The @manipulate macro

9. Additional Exercises

9.1. Exploratory visualization of a temperature dataset

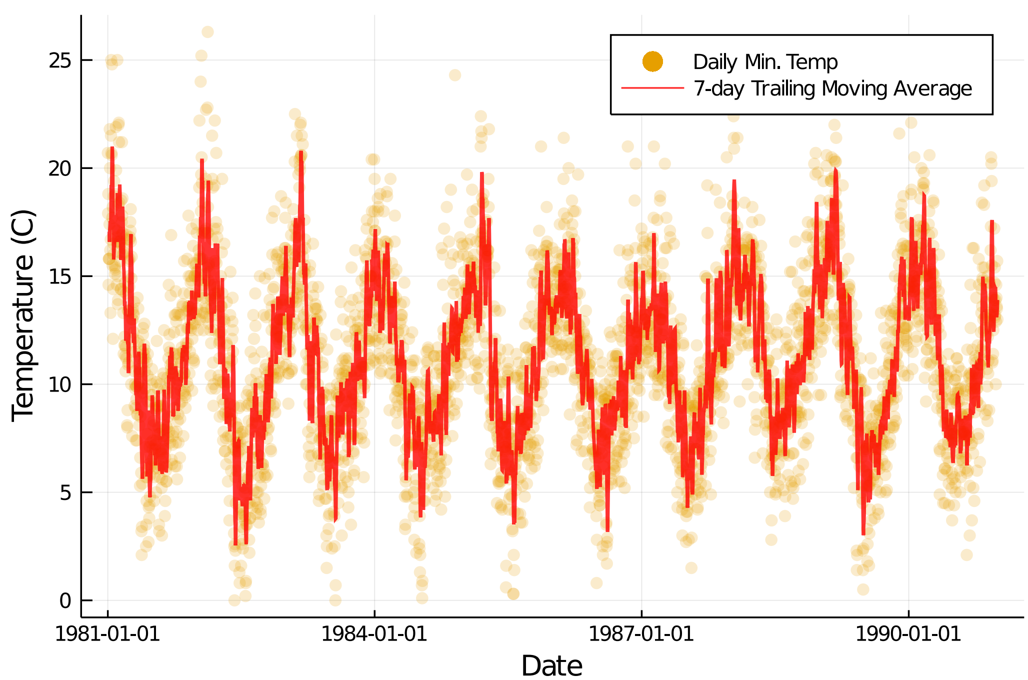
Complete the `??` below to create an interactive plot of temperature and moving average data.

Hint: recall that `df.col` *may be used to index a column named* `col` *in a DataFrame called* `df`*.*

```julia
using Statistics: mean

"Compute the `n`-step moving average for a vector `v`."
moving_average(v::AbstractVector, n::Integer) = [mean(v[(i - n + 1):i]) for i in n:length(v)]

@manipulate for avg_period in (7, 14, 30, 100)
    # scatterplot of raw temperature data vs date
    scatter(
        min_temp.Date, min_temp.Temp;
        alpha=0.2, # partial transparency helps when there are many overlapping points
        xlabel="Date",
        ylabel="Temperature (C)",
        label="Daily Min. Temp"
    )

    # compute and plot moving average
    moving_avg = moving_average(min_temp.Temp, avg_period)
    plot!(
        min_temp.Date[avg_period:end], moving_avg;
        linewidth=2,
        color=:red,
        label="$(avg_period)-day Trailing Moving Average"
    )
end
```

2s

avg_period 7 14 30 100



Comment on the characteristics of the plot.

- Do the data appear to be periodic? If so, what is the approximate length of the period?
- What are the pros and cons of looking at the moving average rather than at the raw data?
- Why does the moving average "lag behind" the raw data, especially when `avg_period` is large?

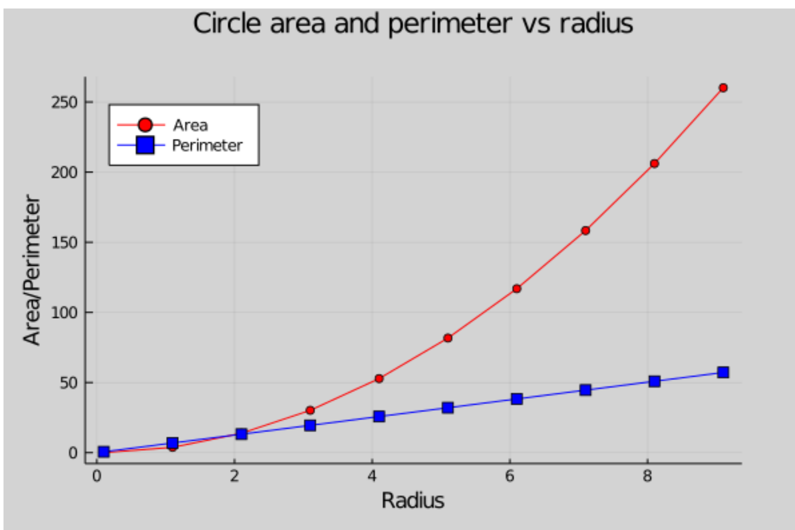
1. Yes, the data appears to be periodic, the approximate length of period is 12 months (one year).

2. The one of advantages of using the moving average is that it can smooth out short-term fluctuations and highlight longer-term trends. On the other hand, it is slower to respond to rapid data, because it gives too much weight to old data.

3. Raw data leads and moving avereage follows because moverage is based on the past data, especially when the avg_period is large, even though the temperature is changed rapidly, the average temperature would not reflect the most recent trends.

9.2. Plotting multiple series

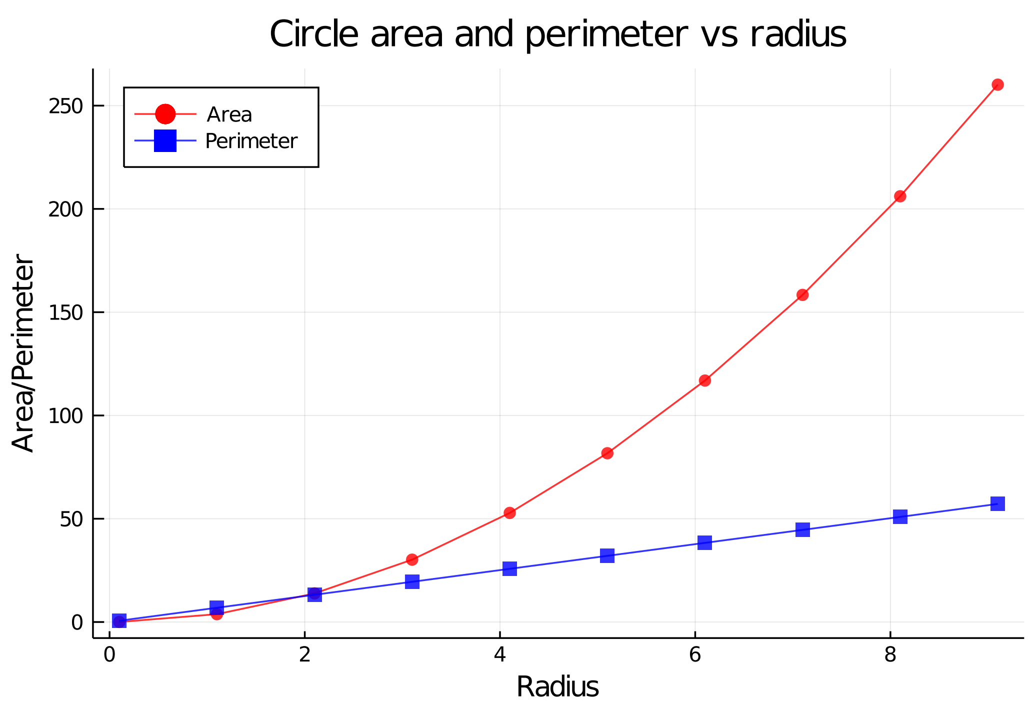
Complete the code below to compute the area and perimeter for a range of circle radius values, then plot the area and perimeter data on a single plot. Consider the following plot as a reference:




```
rr = 0.1:10

aa = [] # area vector
pp = [] # perimeter vector

for r in rr
    a, p = circle_area_perimeter(r)
    push!(aa, a)
    push!(pp, p)
end

area_perimeter_vs_radius = plot(
    rr, aa;
```

```
    color=:red,
    marker=:circle,
    label="Area",
    xlabel="Radius",
    ylabel="Area/Perimeter",
    legend=:topleft,
    title="Circle area and perimeter vs radius"
)
plot!(
    rr, pp;
    color=:blue,
    marker=:square,
    label="Perimeter"
)
```

✓ 1s



```julia
using PlotCheck
@check_plot area_perimeter_vs_radius
```

✓ 2s

PlotCheck Report

Subplot 1

Series 'Area': Checked.

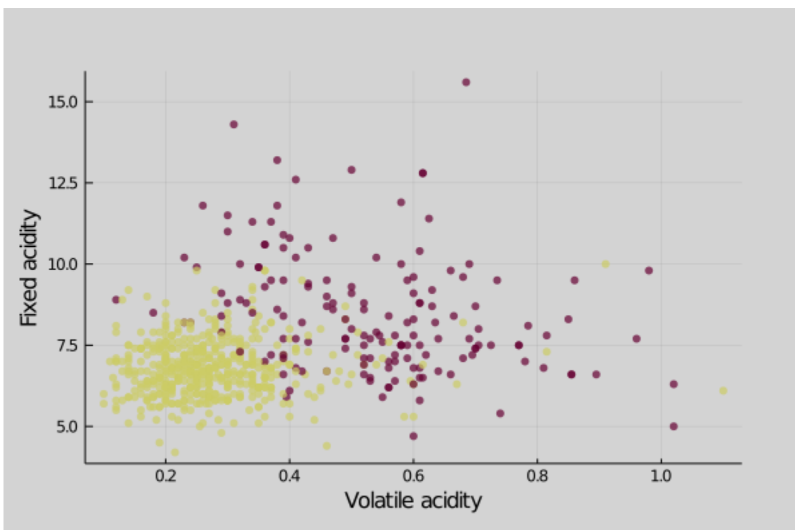
Series 'Perimeter': Checked.

```
Checked against reference plot.
```

9.3. A scatter plot with differently-colored points

Create a scatter plot of fixed acidity versus volatite acidity. Volatile acidity values should be on the x-axis, and fixed acidity values should be on the y-axis. Point color should indicate the type of wine: white or red.

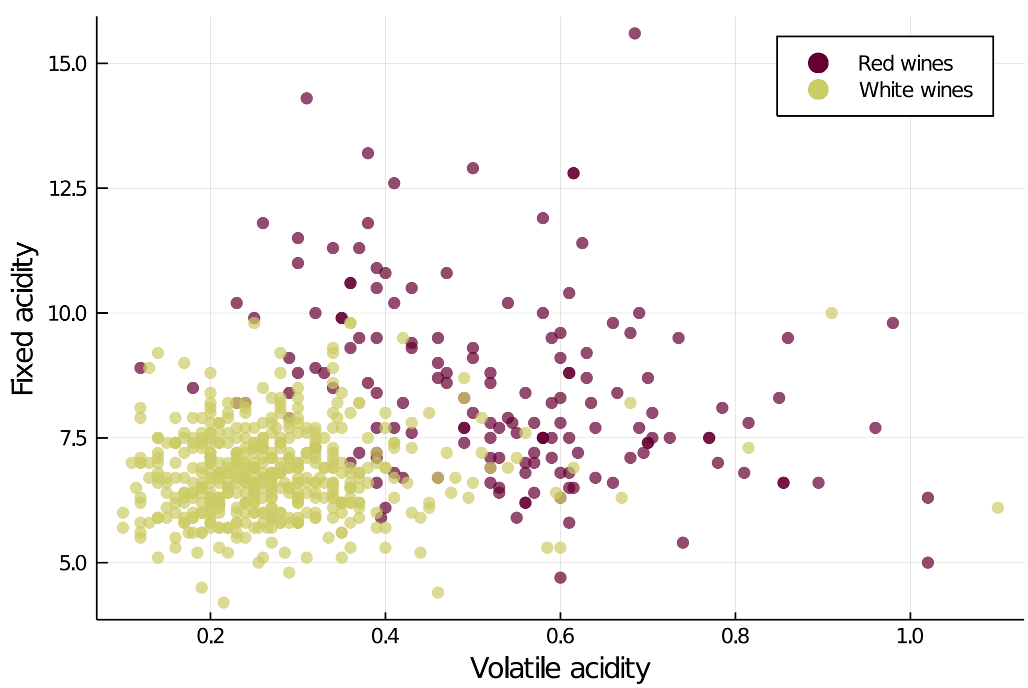
When you're finished, the plot should look like this:




```
using Plots, CSV, DataFrames

default(
    markerstrokewidth=0.3,
    markerstrokecolor=:auto,
    alpha=0.8,
    label=""
)

wine = DataFrame!(CSV.File("wine.csv"))

# Select only the red and white wines respectively
wine_red = wine[wine.Color .== "Red" ,:]
wine_white = wine[wine.Color .== "White" ,:]

# Define appropriate colors to indicate white and red
```

```
c_white = RGB(0.8, 0.8, 0.4)
c_red = RGB(0.4, 0.0, 0.2)

# Plot the red wines
fixedacidity_vs_volatileacidity = scatter(
    wine_red.volatileacidity, wine_red.fixedacidity;
    color=c_red,
    alpha=0.7,
    xlabel="Volatile acidity",
    ylabel="Fixed acidity",
    label="Red wines"
)

# Add the white wines
scatter!(
    wine_white.volatileacidity, wine_white.fixedacidity;
    color=c_white,
    alpha=0.7,
    label="White wines"
)
```

28s



```julia
using PlotCheck
@check_plot fixedacidity_vs_volatileacidity
```

✓ 8s

PlotCheck Report

Subplot 1

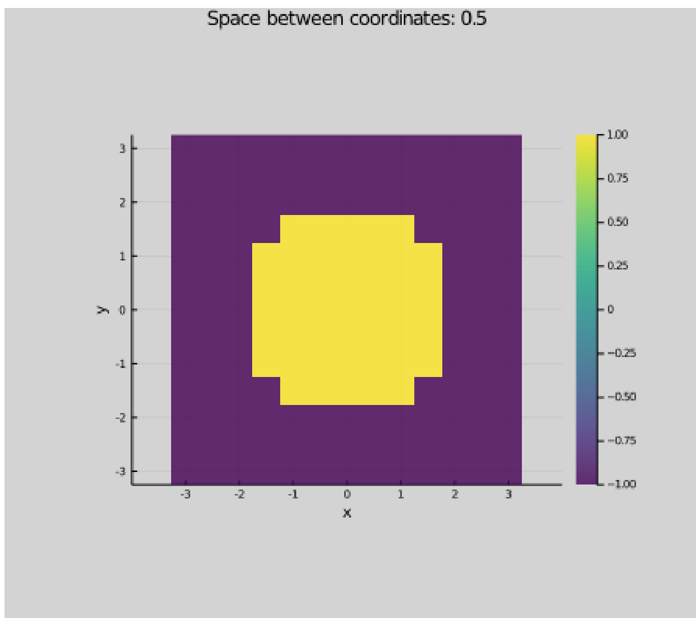
Series 'White wines': Checked.

Series 'Red wines': Checked.

```
Checked against reference plot.
```

9.4. Visualizing decision boundaries

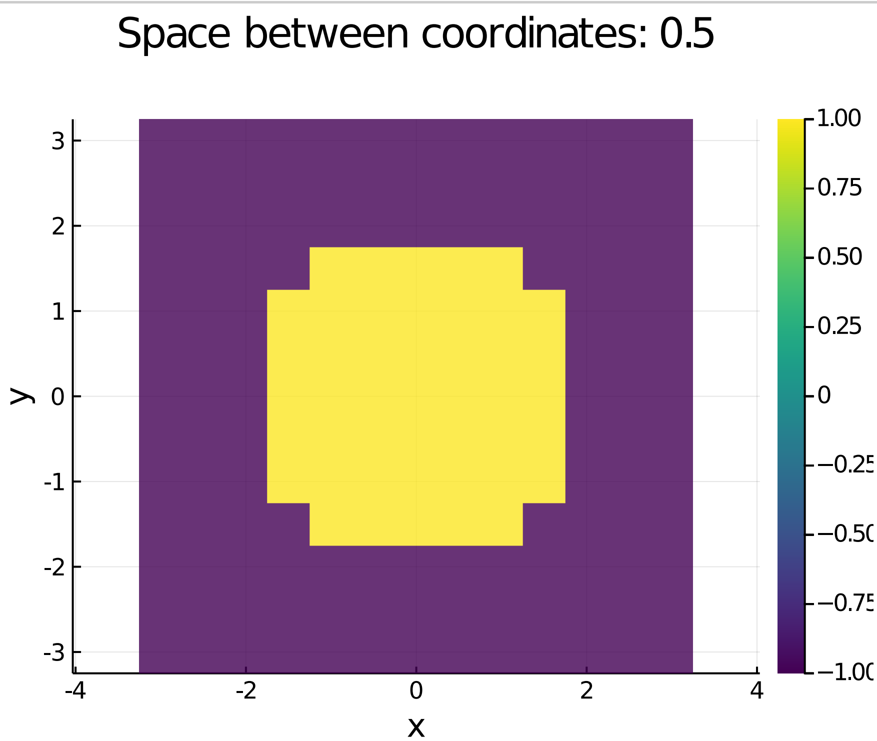
Complete the code below to visualize a circular decision boundary with radius 2 over a grid of points spaced 0.5 units apart. The finished plot should look like this:




```
r = 2.0
s = 0.5
x_range = -3:s:3
y_range = copy(x_range)

D = circular_decision_boundary(r, x_range, y_range)

"Visualize a decision boundary encoded in `D` over a grid of points encoded in `x_range` and `y_range`"
function heatmap_decision_boundary(x_range::AbstractRange, y_range::AbstractRange, D::AbstractMatrix; kwargs...)
    return heatmap(
        x_range, y_range, D;
        aspect_ratio=1.0,
        size=(450, 400),
```

```julia
        xlabel="x",
        ylabel="y",
        kwargs...
        )
end

heatmap_decision_boundary_lowres = heatmap_decision_boundary(
    x_range,
    y_range,
    D;
    title="Space between coordinates: $(s)"
)
```

✓ 1s




```julia
using PlotCheck
@check_plot heatmap_decision_boundary_lowres
```

✓ 81ms

PlotCheck Report

Subplot 1

Series '': Checked.

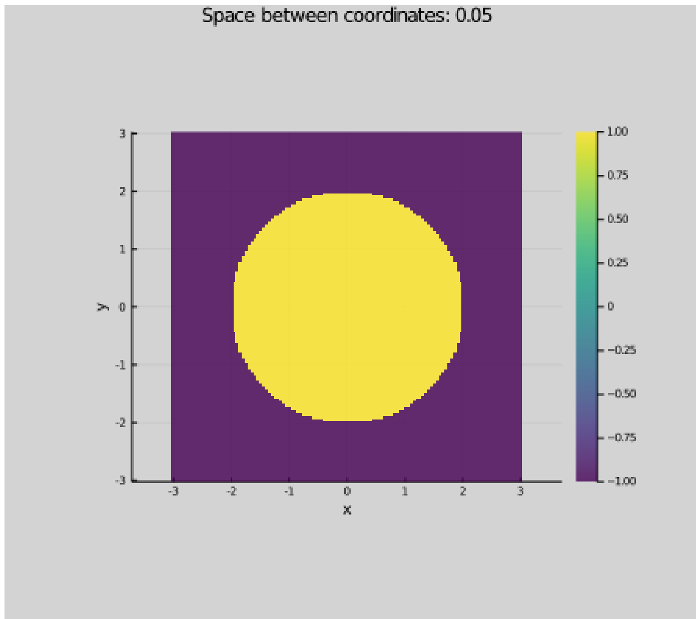
`Checked against reference plot.`

Why didn't we set `yflip=true` in the heatmap command?

Hint: See what happens if you set `yflip=true` and notice what axes is off.

We set yflip=true to let the heatmap upside down and it is exactly looking the same.

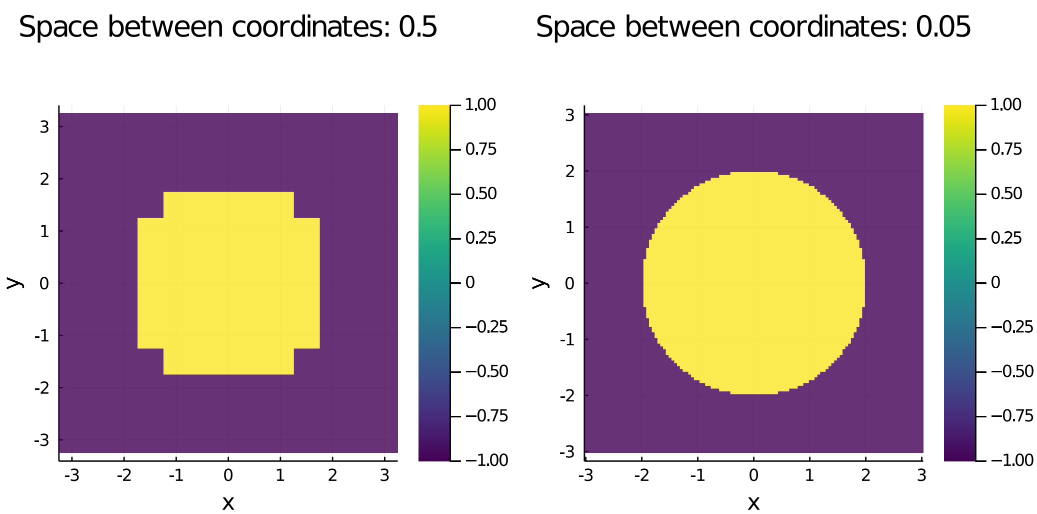
Now increase the point grid resolution by changing the step from 0.5 to 0.05, corresponding to ten times as many points in both dimensions. Visualize this refined decision boundary with another heatmap. This new heatmap should look like the following:



Finally, show both heatmaps next to each other to make comparison easier.

```
r = 2.0
s = 0.05
x_range = -3:0.05:3
y_range = copy(x_range)

D = circular_decision_boundary(r, x_range, y_range)
heatmap_decision_boundary_highres = heatmap_decision_boundary(
    x_range,
    y_range,
    D;
    title="Space between coordinates: $(s)"
)

plot(
    heatmap_decision_boundary_lowres,
    heatmap_decision_boundary_highres;
    layout=(1, 2),
    size=(750, 400)
)
```

365ms




```
1 @check_plot heatmap_decision_boundary_highres
```

86ms

PlotCheck Report

Subplot 1

Series '': Checked.

```
Checked against reference plot.
```

9.5. Programmatically generating a file name

Run the following cell to ensure your function works.

```
prefix = "windspeed"
suffix = "2015-05-10"
index = 5
extension = "tsv"

generated_filename = make_filename(prefix, suffix, index, extension)
target_filename = "windspeed_2015-05-10_5.tsv"
isequal(generated_filename, target_filename) && println("Generated file name matches target filename.")
```

✓ 69ms

```
Generated file name matches target filename.
```

9.6. Downloading files programmatically with `try`/ `catch`

- How many errors were there?
- Which image files were not found in the cloud folder?

There were 2 errors. "Data_file_0.png" and "Data_file_9.png" are not found in the could folder.

```
using Images

img_plots = []
for filename in readdir(folder_local)
    if filename[(end - 2:end)] == "png"
        img = joinpath(folder_local, filename) |> load |> Array
        push!(img_plots, heatmap(img; axis=false, grid=false, aspect_ratio=1.0, size=(30, 30)))
    end
end
plot(img_plots...; layout=(1, length(img_plots)), size=(850, 100))
```

✓ 1s

Completed on 2021-09-03 at 3:09PM
by Yuzhan Jiang