HELIX Rotor Torque Calculation

Julia stuff for easier calculation

In [1]:
```
thisdir = pwd()
```
Out[1]:
```
"/Users/noahgreen/Documents/GitHub/helix-magnetic-field"
```

In [2]:
```
using DataFrames
```

In [3]:
```
using CSV
```

In [4]:
```
struct HelixCoil{U<:Float64,V<:Int64}
    current::U
    rinner::Vector{U}
    router::Vector{U}
    turns::Vector{U}
    divρ::Vector{V}
    divz::Vector{V}
end
```

In [5]:
```
function HelixCoil(dict::Dict)
    current = parse(Float64,dict[:current])
    rinner = parse.(Float64,dict[:inner_radius])
    router = parse.(Float64,dict[:outer_radius])
    turns = parse.(Float64,dict[:subcoil_turns])
    divρ = parse.(Int64,dict[:subcoil_rho_div])
    divz = parse.(Int64,dict[:subcoil_z_div])
    return HelixCoil{Float64,Int64}(current,rinner,router,turns,divρ,divz)
end
```
Out[5]:
```
HelixCoil
```

Load and parse data

Get information from the best fit of the coils to the measured magnetic field

In [6]:
```
helix_config_io = open("helix_config_fitted.csv","r")
helix_config_str = read(helix_config_io,String)
close(helix_config_io)
```

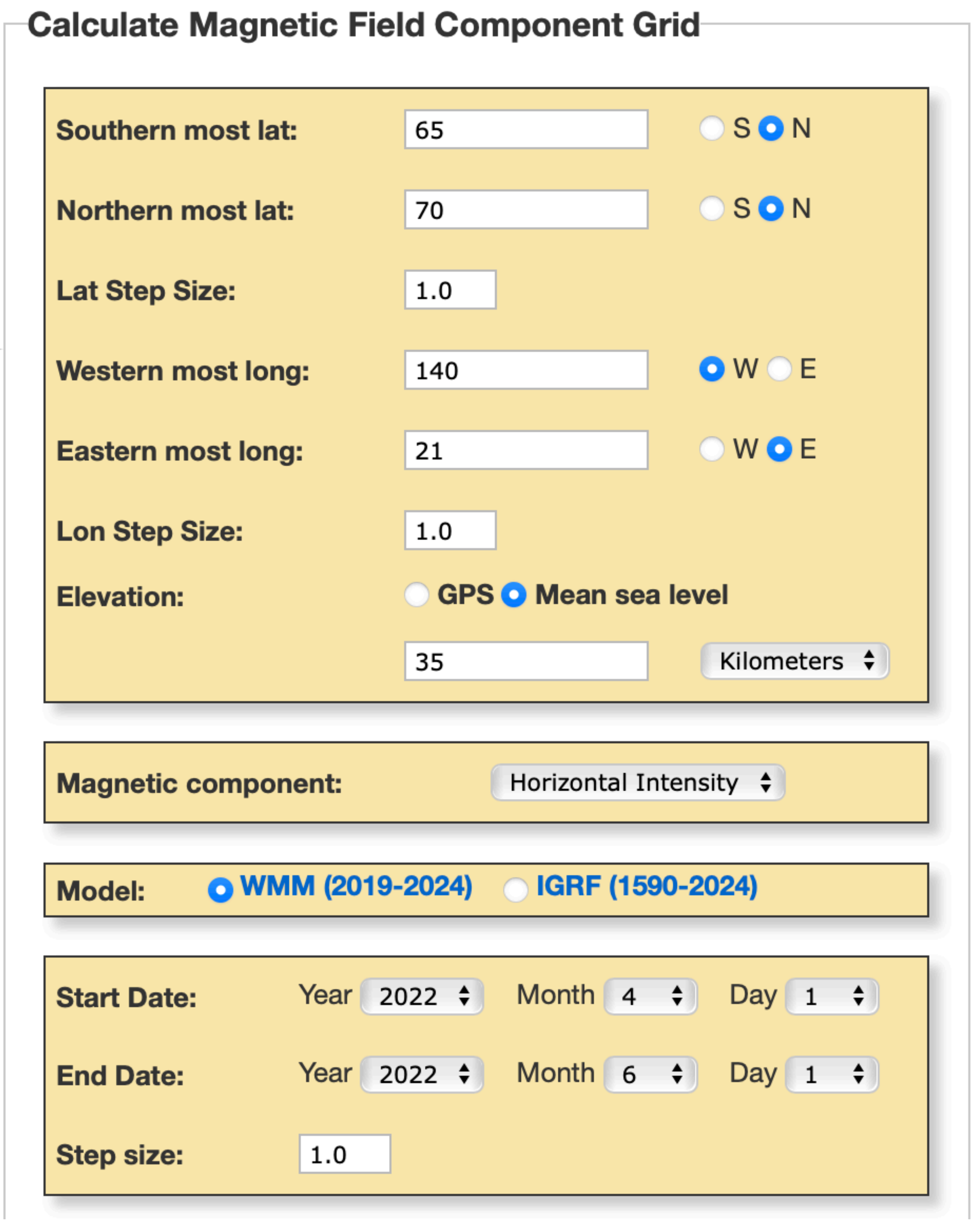
In [7]:
```
helix_config_str_list = split(helix_config_str,"\n")
```
Out[7]:
```
22-element Vector{SubString{String}}:
 "default_contraction,0"
 "new_coil"
 "current,91.5"
 "width,0.075818"
 "rotation_angleaxis,1.570381514159932,0.006311291406169085,0.9999601668074448,-0.006311291406168431"
 "origin,-0.3570241330755666,-0.005845356030904593,5.094722084431961e-05"
 "inner_radius,0.1962736292766629,0.2073585464821345,0.2262812157433628"
 "outer_radius,0.2073585464821345,0.2262812157433628,0.2461403261570631"
 "subcoil_turns,1995.9,5150,5489.5"
 "subcoil_rho_div,8,8,8"
 "subcoil_z_div,52,32,32"
 "new_coil"
 "current,91.5"
 "width,0.075818"
 "rotation_angleaxis,1.570579546934161,0.00151963059283926,0.9999976907201948,-0.001519630592839103"
 "origin,0.3482921329327381,-0.0007180087020030899,-0.005737567305902048"
 "inner_radius,0.1961598448054855,0.2069948489118565,0.2262520443774237"
 "outer_radius,0.2069938849435197,0.2262520443774237,0.2469898952084166"
 "subcoil_turns,1976,5110,5486.7"
 "subcoil_rho_div,8,8,8"
 "subcoil_z_div,52,32,32"
 ""
```

In [8]:
```
coil_dict_vec = Dict[]
coil_dict = nothing
for row in helix_config_str_list
    rowlist = split(row,",")
    if rowlist[1] == "new_coil"
        if coil_dict === nothing
            coil_dict = Dict()
            continue
        else
            push!(coil_dict_vec,coil_dict)
            coil_dict = Dict()
            continue
        end
    elseif rowlist[1] == "current"
        coil_dict[:current] = string(rowlist[2])
    elseif rowlist[1] == "inner_radius"
        coil_dict[:inner_radius] = string.(rowlist[2:end])
    elseif rowlist[1] == "outer_radius"
        coil_dict[:outer_radius] = string.(rowlist[2:end])
    elseif rowlist[1] == "subcoil_turns"
        coil_dict[:subcoil_turns] = string.(rowlist[2:end])
    elseif rowlist[1] == "subcoil_rho_div"
        coil_dict[:subcoil_rho_div] = string.(rowlist[2:end])
    elseif rowlist[1] == "subcoil_z_div"
        coil_dict[:subcoil_z_div] = string.(rowlist[2:end])
    else
        continue
    end
end
push!(coil_dict_vec,coil_dict)
```
Out[8]:
```
2-element Vector{Dict}:
 Dict{Any, Any}(:inner_radius => ["0.1962736292766629", "0.2073585464821345", "0.2262812157433628"], :subcoil_turns => ["1995.9", "5150", "5489.5"], :subcoil_rho_div =>
["8", "8", "8"], :subcoil_z_div => ["52", "32", "32"], :outer_radius => ["0.2073585464821345", "0.2262812157433628", "0.2461403261570631"], :current => "91.5")
 Dict{Any, Any}(:inner_radius => ["0.1961598448054855", "0.2069948489118565", "0.2262520443774237"], :subcoil_turns => ["1976", "5110", "5486.7"], :subcoil_rho_div => [
"8", "8", "8"], :subcoil_z_div => ["52", "32", "32"], :outer_radius => ["0.2069938849435197", "0.2262520443774237", "0.2469898952084166"], :current => "91.5")
```

In [9]:
```
helix_coils = HelixCoil.(coil_dict_vec)
```
Out[9]:
```
2-element Vector{HelixCoil{Float64, Int64}}:
 HelixCoil{Float64, Int64}(91.5, [0.1962736292766629, 0.2073585464821345, 0.2262812157433628], [0.2073585464821345, 0.2262812157433628, 0.2461403261570631], [1995.9, 51
50.0, 5489.5], [8, 8, 8], [52, 32, 32])
 HelixCoil{Float64, Int64}(91.5, [0.1961598448054855, 0.2069948489118565, 0.2262520443774237], [0.2069938849435197, 0.2262520443774237, 0.2469898952084166], [1976.0, 51
10.0, 5486.7], [8, 8, 8], [52, 32, 32])
```

Get information for Earth's B-field at time and location of flight

Information acquired from the NOAA's world magnetic model. I only take the horizontal component of the field since the vertical component does not contribute to vertical torque. Screenshot of query below:



In [11]:
```
wmmSwedenFlight2022 = DataFrame(CSV.File(joinpath(thisdir,"igrfgridData.csv")))
```
Out[11]: 972 rows × 7 columns (omitted printing of 1 columns)

	date	latitude(deg)	longitude(deg)	elevation(km)	horintensity(nT)	annualchange(nT)
	Float64	Float64	Float64	Float64	Float64	Float64
1	2022.25	70.0	-140.0	35.0	8404.9	39.0
2	2022.25	70.0	-139.0	35.0	8300.0	40.1
3	2022.25	70.0	-138.0	35.0	8193.8	41.2
4	2022.25	70.0	-137.0	35.0	8086.3	42.4
5	2022.25	70.0	-136.0	35.0	7977.6	43.5
6	2022.25	70.0	-135.0	35.0	7867.7	44.6
7	2022.25	70.0	-134.0	35.0	7756.9	45.8
8	2022.25	70.0	-133.0	35.0	7645.2	46.9
9	2022.25	70.0	-132.0	35.0	7532.7	48.0
10	2022.25	70.0	-131.0	35.0	7419.4	49.2
11	2022.25	70.0	-130.0	35.0	7305.6	50.3
12	2022.25	70.0	-129.0	35.0	7191.4	51.4
13	2022.25	70.0	-128.0	35.0	7076.8	52.5
14	2022.25	70.0	-127.0	35.0	6962.0	53.6
15	2022.25	70.0	-126.0	35.0	6847.2	54.7
16	2022.25	70.0	-125.0	35.0	6732.4	55.8
17	2022.25	70.0	-124.0	35.0	6618.0	56.9
18	2022.25	70.0	-123.0	35.0	6504.0	58.0
19	2022.25	70.0	-122.0	35.0	6390.6	59.0
20	2022.25	70.0	-121.0	35.0	6278.1	60.0
21	2022.25	70.0	-120.0	35.0	6166.5	61.0
22	2022.25	70.0	-119.0	35.0	6056.3	62.0
23	2022.25	70.0	-118.0	35.0	5947.5	62.9
24	2022.25	70.0	-117.0	35.0	5840.5	63.8
25	2022.25	70.0	-116.0	35.0	5735.4	64.7
26	2022.25	70.0	-115.0	35.0	5632.6	65.5
27	2022.25	70.0	-114.0	35.0	5532.3	66.3
28	2022.25	70.0	-113.0	35.0	5434.9	67.0
29	2022.25	70.0	-112.0	35.0	5340.5	67.6
30	2022.25	70.0	-111.0	35.0	5249.7	68.2
⋮	⋮	⋮	⋮	⋮	⋮	⋮

Calculate torque

In [18]:
```
"""
    μ(coil::HelixCoil)

Returns the magnitude of the magnetic moment of one of the two coils of the HELIX magnet
"""
function μ(coil::HelixCoil)
    # The each coil is broken into 3 subcoils for which the outer and inner radius is defined.
    # Each subcoil is composed of many current loops, each of which have a magnetic moment of μ=IA oriented perpendicular to the current flow.

    # Use the average area of each subcoil as the area of their loops
    ainner = π*coil.rinner .^2
    aouter = π*coil.router .^2
    aavg = (ainner + aouter)/2

    # Since the current is the same in all the loops, we can combine their areas into an effective area for the subcoil
    effective_area = coil.turns .* aavg

    # μ_subcoil=IA
    μsubcoils = coil.current*effective_area

    # μ_coil = Σ μ_subcoil
    return sum(μsubcoils)
end
```
Out[18]:
```
μ
```

In [19]:
```
# and the magnetic moment for each coil is...
μHelixCoils = μ.(helix_coils)
```
Out[19]:
```
2-element Vector{Float64}:
 181313.13937894936
 180637.4135329068
```

In [20]:
```
# extract the maximum horizontal B-field for the max torque
maxB = max(wmmSwedenFlight2022."horintensity(nT)"...)
```
Out[20]:
```
12822.6
```

In [22]:
```
# convert to Teslas
maxBtesla = maxB*1e-9
```
Out[22]:
```
1.2822600000000002e-5
```

In [23]:
```
# τ = μ x B
max_torque_nm = sum(μHelixCoils)*maxBtesla
```
Out[23]:
```
4.641147159767567
```

In [24]:
```
# convert to ft-lbs for us Americans...
max_torque_ftlbs = max_torque_nm*(1/0.3048)*(1/4.448)
```
Out[24]:
```
3.4233050270666094
```

Hence, the maximum torque due to the magnet interacting with the Earth's magnetic field is $\tau = 3.42\ \text{ft-lbs}$. For an engineering tolerance though, let's get the torque in a 0.5 gauss B-field:

In [26]:
```
eng_max_torque = max_torque_ftlbs*0.5e-4/maxBtesla
```
Out[26]:
```
13.348716434524237
```

In [ ]: