Topic 1.

origin ACFlow barycentric works bad on finding poles except the first pole.

One solution is to find poles one by one and remove poles that have been found. I code and find it works.

```julia
#Examples of find poles by ACFlow barycenteric

using DelimitedFiles
using Printf
using ACFlow
using Plots
include("Method.jl")



function iter_solve(β,poles,input_G)

    B = Dict{String,Any}(
        "solver" => "BarRat",  # Choose MaxEnt solver
        "mtype"  => "gauss",   # Default model function
        "mesh"   => "tangent", # Mesh for spectral function
        "ngrid"  => length(poles),        # Number of grid points for input data
        "nmesh"  => 801,       # Number of mesh points for output data
        "wmax"   => 8.0,       # Right boundary of mesh
        "wmin"   => -8.0,      # Left boundary of mesh
        "beta"   => β,      # Inverse temperature
    );

    S = Dict{String,Any}(
        "atype"=>"delta",
        "denoise"=>"none",
        "epsilon"=>1e-10,
        "pcut"=>0.99,
        "eta"=>1e-2,
    );
    setup_param(B, S);
    solve(poles,input_G)
end


#descrete situation
β=10.0;
N=20;
poles=(collect(0:N-1).+0.5)*2π/β;
grid=im*poles;
```

```
input_G=Vector{ComplexF64}(undef,N);
for i=1:N
    for j=1:N
        input_G[i]+=1/(grid[i]-poles[j])
    end
end

reA=Vector{Float64}(undef,N);

for k=1:N
    poles1=poles[k:N]
    input_G1=input_G[k:N]
    iter_solve(β, poles1, input_G1);
    input_G=input_G-1 ./(im*poles.-poles[k])
end

#= In each iteration, the first pole we find is:
0.31326707887540683 - 0.001976798372050 32im
0.9502968585820134 + 0.00132689950131284im
1.5448689080539983 + 0.009367377210435299im
2.171973557562636 - 0.006370050599299192im
2.8115280708244406 - 0.06490981203494114im
3.4508359231381087 - 0.061767448739761435im
4.090475826851209 - 0.05849391185379163im
4.792559928441718 - 0.11475243020387278im
5.423083907019795 - 0.09845070549608285im
6.049810676089182 - 0.08194271090056918im
6.670425403533103 - 0.06689387175141867im
7.436589140827179 - 0.10605922293040414im
8.036649263408075 - 0.08103145347725821im
...

which is close to the real poles as follows:
=#
println(poles)
```

Then I will try find the theory reason and try to find and prove a method that can always find the nearst pole with high accuracy. After mu learning and use of AD.

Topic 2.

In the question to do analytic continuation, are poles are known in advance?

If it does, how accuracy can we get when we calculate γ of

$$A = \sum_k \gamma_k \delta(x - x_k)$$

Set

$$C = \left(\frac{1}{ix_j - x_k}\right)_{jk}, \ \bar{G} = (G(ix_j))_j$$

Then $\delta = (\delta_1, .., \delta_n)$ is the solution of

$$C\delta = \bar{G}$$

However, by svd we find that the max and min singular values of C have a significant difference and the solution of this system of equations is numerically extremely unstable.

Process like topic one doesn't work. I have find some methods to get high accurate results when N=10. this is a purely solving equation problem and I would like to postpone addressing this issue.

Topic 3.

How to analysis the erroe and coverage rate of these method and add AD ?

It's difficult and I have no idea.

Topic 4.

Read the aaa algorithm's paper and learn from the examples and math method from it. Try to write general methods to find poles and analise them.