paths

```
dirDataLocker = dirIO <> "rcs/tools/data/";
```

```
sciaccarcs = "rcs-values.dat";
sciaccaelevation = "elevation.dat";
```

```
sciaccaMesh = Range[-180, 180];
```

linear algebra

```
Clear[BuildAFourierCos];
BuildAFourierCos[mesh_List, degree_Integer] := Module[{λ, A, one, vector},
   λ = Length[mesh];
   one = Table[1, {λ}];
   A = {one};
   vector = 1;
   Do[
    vector = vector + mesh;
    AppendTo[A, Cos[vector π/180]]
    , {k, degree}];
   Return[A^T];
  ];
```

```
Clear[basisFourierCos];
basisFourierCos[θ_, degree_Integer] := Module[{},
  Return[Table[Cos[k θ], {k, 0, degree}]];
 ]
```

```
Clear[κ];
κ[A_List] := Module[{s},
  s = SingularValueList[A];
  Return[First[s]/Last[s]];
 ]
```

error

```
Clear[error];
error[A_List, solution_List, data_List] :=
 Module[{σ, m, n, W, Winv, signalToNoise},
  m = Length[data];
  n = Length[solution];
  residual = A.solution - data;
  totalError = residual.residual;
  W = Aᴴ.A;
  Winv = Inverse[W];
  σ = Sqrt[totalError/(m - n) Diagonal[Winv]];
  signalToNoise = Abs[solution]/σ;

  Return[{σ, signalToNoise}]
 ]
```

```
Clear[errorN];
errorN[A_List, solution_List, data_List] :=
 Module[{σ, m, n, W, Winv, signalToNoise},
  m = Length[data];
  n = Length[solution];
  residual = A.solution - data;
  totalError = residual.residual;
  W = Aᴴ.A // N;
  Winv = Inverse[W];
  σ = Sqrt[totalError/(m - n) Diagonal[Winv]];
  signalToNoise = Abs[solution]/σ;
  Return[{σ, signalToNoise}]
 ]
```

```
Clear[printError];
printError[solutionVector_List, errorVector_List] := Module[{m, n, W, Winv},
  Print[totalError, ": r.r = total squared error (totalError)"];
  Print[solutionVector, ": solution vector"];
  Print[errorVector, ": error vector (s)"];
  Print[Abs[solutionVector] / errorVector, ": signal to noise (signalToNoise)"];
  Print[residual, ": residual error vector (residual)"];
 ]
```

parameters

```
(* OTH frequency range *)
ν = Range[3, 30];
```

Out[•]= {3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15,
16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30}

```
(* remap frequencies to [-1, 1] *)
μ = (2/27 # - 11/9) & /@ ν;
```

Out[•]= $\left\{-1, -\frac{25}{27}, -\frac{23}{27}, -\frac{7}{9}, -\frac{19}{27}, -\frac{17}{27}, -\frac{5}{9}, -\frac{13}{27}, -\frac{11}{27}, -\frac{1}{3}, -\frac{7}{27}, -\frac{5}{27}, -\frac{1}{9}, -\frac{1}{27}, \frac{1}{27}, \frac{1}{9}, \frac{5}{27}, \frac{7}{27}, \frac{1}{3}, \frac{11}{27}, \frac{13}{27}, \frac{5}{9}, \frac{17}{27}, \frac{19}{27}, \frac{7}{9}, \frac{23}{27}, \frac{25}{27}, 1\right\}$

In[•]:=
```
(* speed of light in vacuo *)
c = 299 792 458;
```

comparisons

utilities

```
clear[ClearAll] := Module[{},
   (* clear all variable names and assignments *)
   Clear["Global`*"]; ClearAll["Global`*"]; Remove["Global`*"];
   ClearSystemCache[];
  ];
```

```
Clear[cleanStreams];
cleanStreams[] := Close[#] & /@ Drop[Streams[], 2]
```

---

end