```
/// <summary>
/// Find the 5 most prominent frequencies from magnitude of inData
/// </summary>
/// <param name="inData">input data</param>
/// <param name="sampFreq">sampling frequency in [kHz]</param>
/// <param name="freq">[output] frequencies of magnitudes returned,
/// <returns>prominent magnitudes, sorted in descending order</retur
```

ILNumerics uses distinct array types for return values, input- and output parameters.
No ref/out keywords are used.

```
  ILRetArray<double> FreqPeaks(ILInArray<double> inData, ILOutArray<double> freq = null, double sampFreq = 44.1) {
[ ILRetCell ]                [ ILInCell ]              [ ILOutCell ]
[ ILRetLogical ]             [ ILInLogical ]           [ ILOutLogical ]
```

A `using()` block around the function body handles all input parameters and controls their lifetimes.

```
    using (ILScope.Enter(inData)) {

        ILArray<double> Data = check(inData);
        ILArray<double> retLength = min(ceil(Data.Length / 2
        ILArray<double> Window = stdWindowFunc(Data.Length)
        ILArray<double> magnitudes = abs(fft(Data * Window)
        magnitudes = magnitudes[r(0,end / 2 + 1)];

        ILArray<double> indices = empty();
        ILArray<double> sorted = sort(magnitudes, indices, descending:true);
        if (!isnull(freq))
            freq.a = (sampFreq / 2.0 / magnitudes.Length * indices)[r(0,retLength-1)];
        return magnitudes[r(0,retLength-1)];
    }
}
```

The `.a` property is used for assigning to output parameters.

Not supported for arrays:

C# `var` keyword: ~~`var A =..`~~

Compound operators: `A[0]` ~~`+= 1.0;`~~