Appendix A

Gapmap Algorithm

This algorithm to find Δ from a given spectrum was written by Kristine Lang and subsequently extensively revised by me.

Main Gapmap Algorithm

The gapmap program calls a function `FindPeakInSpectrum` on each of the 128 × 128 pixels individially. `FindPeakInSpectrum` walks out from zero energy at the center of the gap and looks for the first feature it can identify as a coherence peak, subject to several filters described in the next section.

Generally the program is set to look for a peak in the range `en1 = 14 meV` to `en2 = 76 meV`. Then it looks for a peak in the range `en1 = -14 meV` to `en2 = -76 meV`. The results Δ_+ and Δ_- are averaged together to give the final output `Δ`.

A summary of the algorithm is as follows:

```
FindPeakInSpectrum{inSpectrum}

  nPts = size(inSpectrum, /N_ELEMENTS)

  maxVal = MAX(inSpectrum[en1:en2], maxValIndex)

  peakIndex = -1

  posSpectrum = inSpectrum[en1:*]
  derivSpectrum = CalcSimpleDeriv(inSpectrum)[en1:*]
  decrease = (derivSpectrum lt 0)
```

```
fiveptsum = decrease $
        + [decrease[1:*],0] $
        + [decrease[2:*],0,0] $
        + [decrease[3:*],0,0,0] $
        + [decrease[4:*],0,0,0,0]

; indices of points which satisfy 3/5 criterion
threeoffive = where((fiveptsum[0:(en2-en1)] ge 3), count)

; if count<=0 then no peak was found between en1 and en2,
; so return maxValIndex if we suspect the peak is just higher
;   than the given range
; or return error code -1 if we don't think the peak is out of range
if (count le 0) then begin
  if outrangefil then peakIndex=maxValIndex+en1 else peakIndex=-1

; if we are not skipping resonances, then we just return the local max
; by the first point satisfying the 3of5 criterion
end else if (not skipresfil) then begin
  peakIndex = threeoffive[0]
  while ((peakIndex+1) le (en2-en1)) do begin
    if (posSpectrum[peakIndex] ge posSpectrum[peakIndex+1]) then $
      break $
    else $
      peakIndex = peakIndex + 1
  endwhile
  peakIndex = peakIndex + en1

; if we are skipping resonances, then find the first peak which
;   satisfies the skiprespct criterion
; if such points exist, then we return the local max around the first one
; but if such points don't exist, we consider outrangefil
end else begin
  peakIndex = threeoffive[0]
  for i = 0, count-1 do begin
    tempIndex = threeoffive[i]
    while ((tempIndex+1) le (en2-en1)) do begin
      if (posSpectrum[tempIndex] ge posSpectrum[tempIndex+1]) then break $
      else tempIndex = tempIndex + 1
    end
    if (posSpectrum[tempIndex] gt posSpectrum[peakIndex]) then $
      peakIndex = tempIndex
    if (posSpectrum[peakIndex] ge skiprespct/100.0 * maxVal) then break
  end
```

```
      if (posSpectrum[peakIndex] ge skiprespct/100.0 * maxVal) then begin
        peakIndex = peakIndex + en1
      end else begin

        ; if outrangefil is true then we see if the peakIndex we found
        ;   (which is the local max by the global max of all pts satisfying
        ;   the 3of5 criterion) satisfies the outrangepct criterion
        ;   (which is usually less stringent than the outrangecriterion)
        ;   if so, then return peakIndex
        ;   if not, then return maxValIndex
        ; if outrangefil is false, then return -1
        if outrangefil then begin
          if (posSpectrum[peakIndex] ge outrangepct/100.0 * maxVal) then $
            peakIndex = peakIndex + en1 $
          else $
            peakIndex = maxValIndex + en1
        end else begin
          peakIndex = peakIndex + en1
        end
      end ; no peaks found above skiprespct * maxVal
    end ; (skipresfil)
  end ; (en1<en2)
```

Additional Gapmap Filters

`Skip resonances:` If the peak found is less than `x%` of the maximum value in the range, then keep looking for higher peaks, and return the first one to meet the `x%` criterion. If no such peak is found, and if `Peak out of range` is not set, then return `0` (error code).

`Peak out of range:` If no satisfactory peak is found, set gap energy to the location of maximum value in the range, which should be within `2` energy steps of the maximum energy in the range. If `Skip resonances` is set, the peak must be at least `y%` of the maximum value in the range; otherwise any peak is fine.

`Max gap:` Set equal to $\mathtt{MAX}\{\Delta_+, \Delta_-\}$ instead of $(\Delta_+ + \Delta_-)/2$.

`Bad pixel:` Wherever there is a bad pixel (i.e. no coherence peak found) set that pixel equal to the average of its nearest neighbors.

`Contiguity:` If a pixel is not within threshold $\mathtt{T}_1$ of half of its nearest neighbors (i.e. `2 of 4`, `1 of 3`, `1 of 2`), then set that pixel equal to the average of its nearest neighbors.

`Pos/Neg Difference:` If Δ_+ and Δ_- differ by more than threshold T_2, then see which one fits better with its nearest neighbors, set the average gapmap equal to that one. If neither agree with nearest neighbors, set the average gapmap equal to $(\Delta_+ + \Delta_-)/2$.

All gapmaps shown in this thesis were calculated using the `Skip resonances` filter (with $x = 90\%$), `Peak out of range` filter (with $y = 80\%$), and `Pos/Neg Difference` filter (with $T_2 = 10$ meV). It is particularly important to use the `Skip resonances` filter in the Ni-substituted samples. Otherwise the gapmap algorithm will misinterpret the +18 meV resonance as a coherence peak and report an erroneously low value for Δ.