Electronic Supplementary Material (ESI) for Nanoscale.

Supplementary Materials of "A modern look at a medieval bilayer metal leaf: nano-tomography of Zwischgold"

S1: Supporting information regarding modern Zwischgold samples

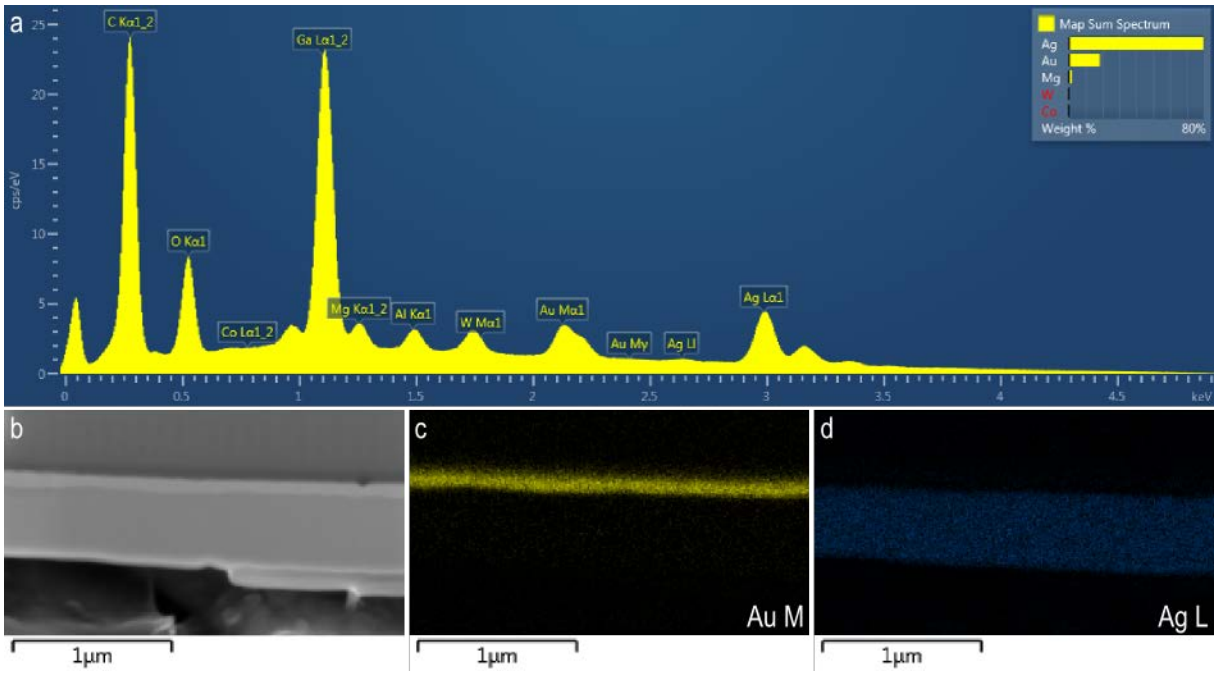

Fig. S 1: a) EDX spectrum of a relatively well-preserved region of the *35-year* sample, imaged by b) SEM and c) Au M_α and d) Ag L_α EDX mapping. Note that C and Al were deposited to protect the sample from ion-deposition during FIB cross-sectioning with a Ga ion beam. A presence of Cu would be observed by the strong Cu L_α and L_β peaks at 930 and 950 eV, just below the weak Ga L_l peak at 957 eV and strong L_α peak at 1098 eV. Sub-panel (b) reproduced from [Wu 2018].

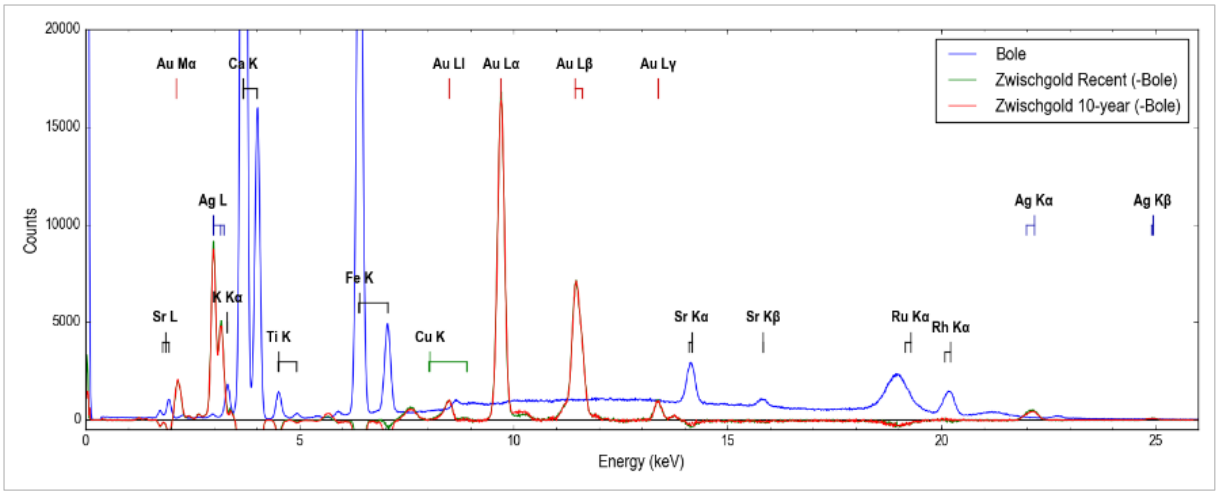

Fig. S 2: X-ray fluorescence spectra of *Recent* (green) and *10-year* (red) Zwischgold and their bole substrates (blue). The bole substrate spectrum has been subtracted from the spectra of the applied Zwischgold leaves. All peaks in the metal leaf spectra can be attributed to Au and Ag, while a significant presence of other metals such as Cu can be excluded.

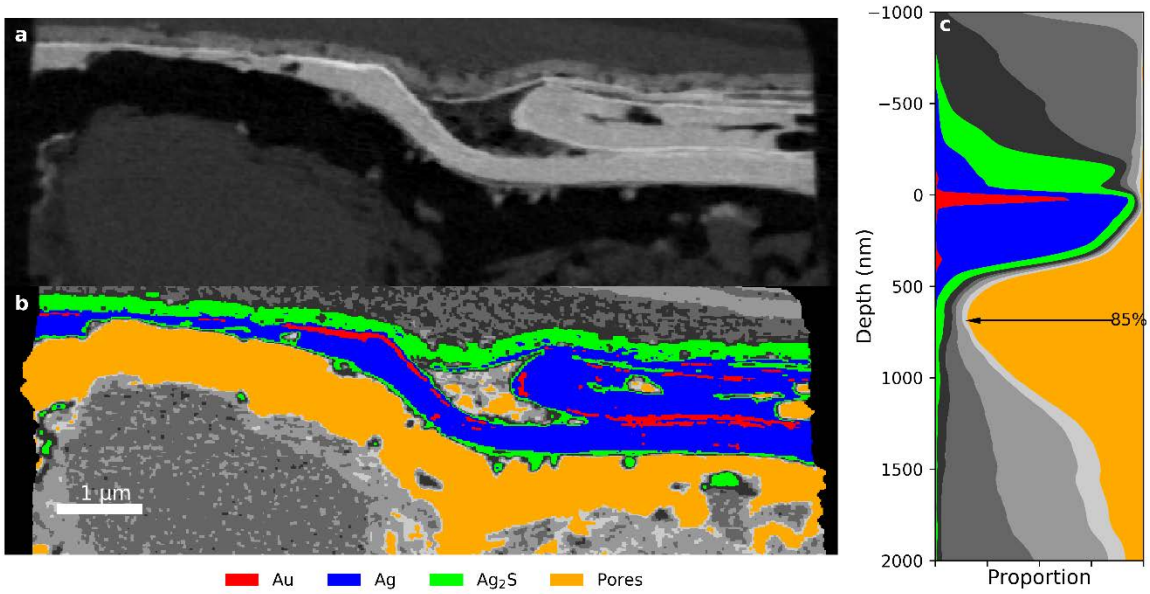

Fig. S 3: Slice through the *35-year* PXCT tomogram in (a) δ-values and (b) segmented according to the ranges shown in Fig. 3 in the main text. (c) Depth profile of the single-layered section of the sample, aligned to the main gold-rich layer.

S2: Supporting information regarding the *Mary* sample

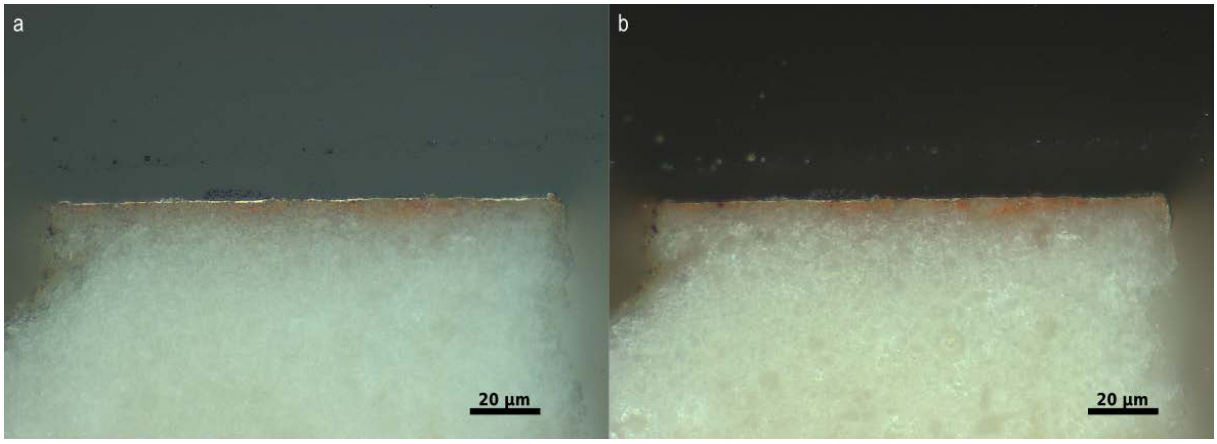

Fig. S 4: a) Bright field and b) dark field visible light microscopy (50x magnification) cross-section images of a sample taken from an adjacent region to the *Mary* sample.

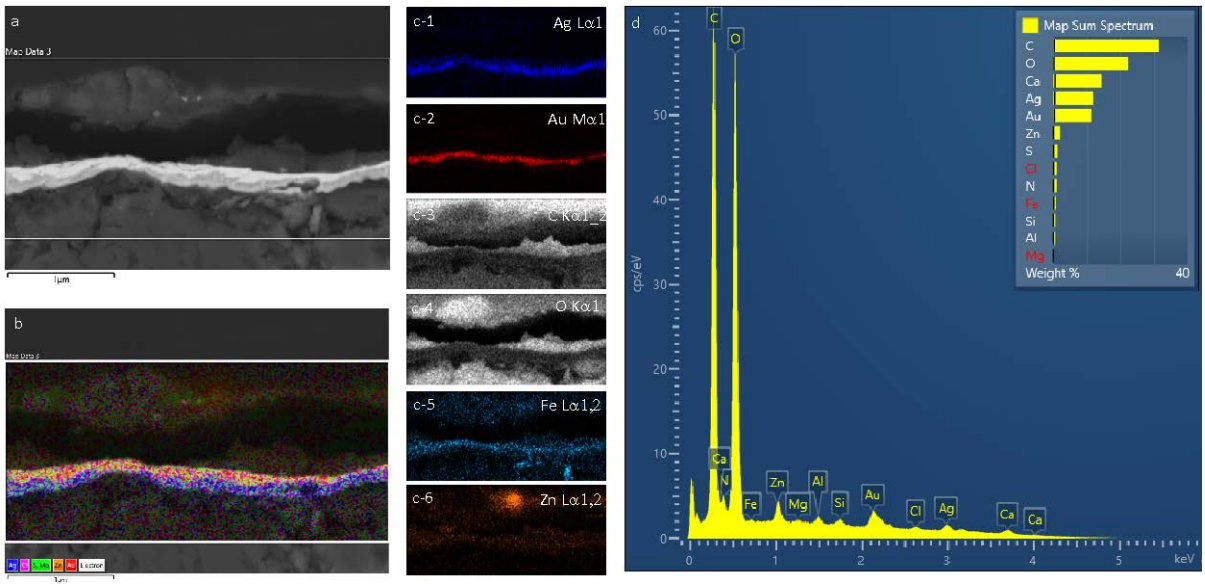

Fig. S 5: SEM-EDX measurements on the same sample cross-section in Fig. S 4 (adjacent to the *Mary* sample): (a) SEM-BSE image; (b) overlay of BSE image and EDX elemental maps; (c1-6): EDX elemental maps for Ag, Au, C, O, Fe and Zn; (d) EDX sum spectrum of the observation site.

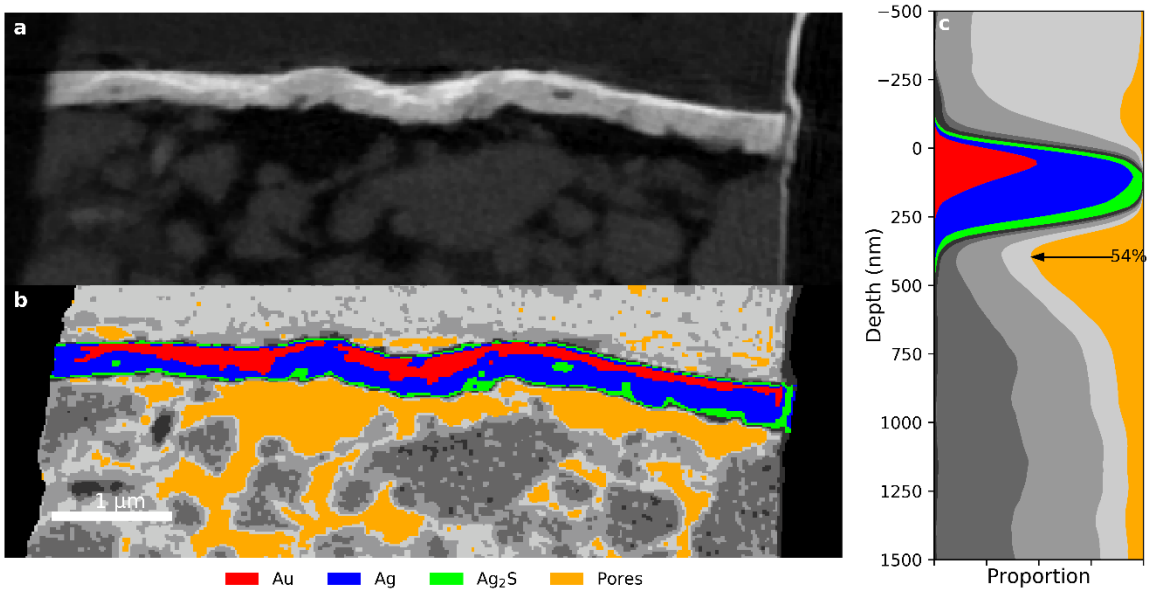

Fig. S 6: Slice through the *Mary* PXCT tomogram in (a) δ-values and (b) segmented according to the ranges shown in Fig. 6 in the main text. (c) Depth profile of the sample, aligned to the main gold-rich layer.

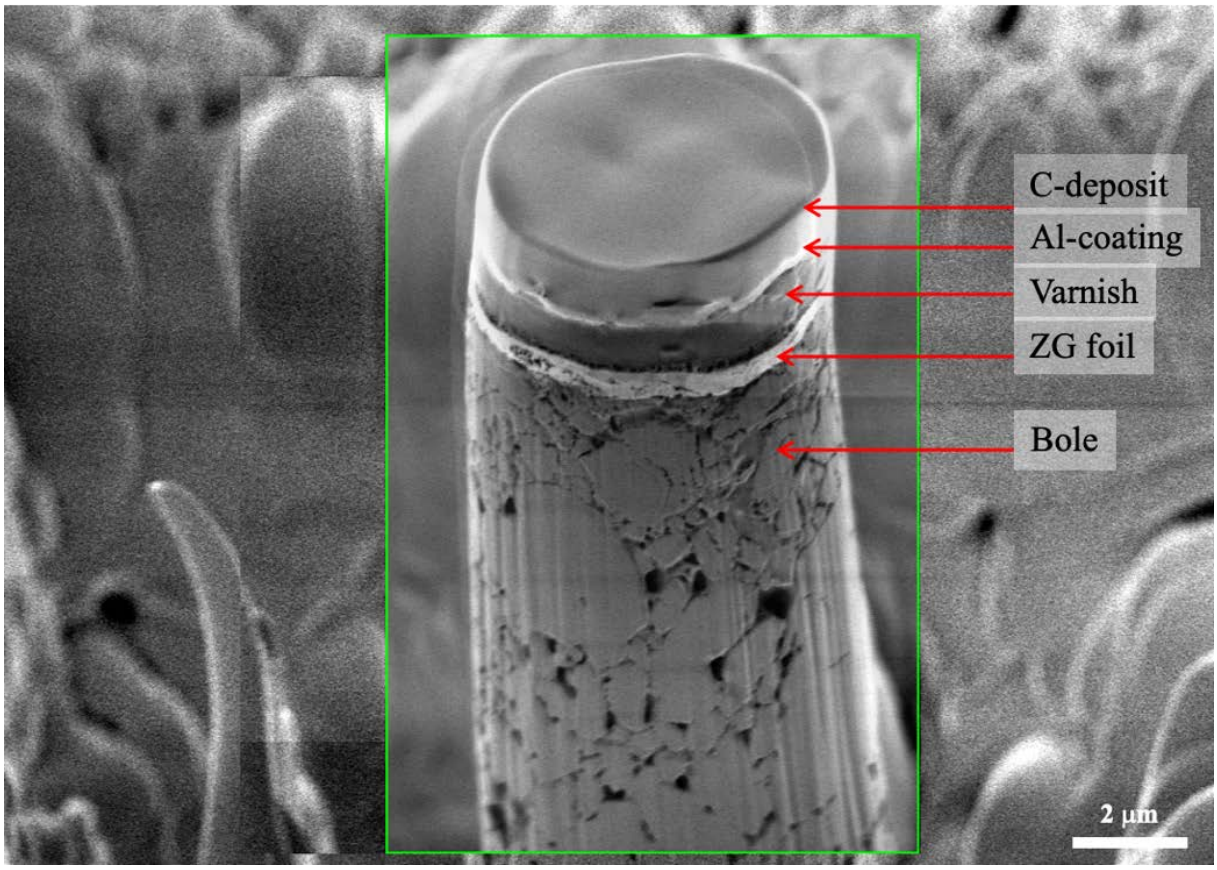

Fig. S 7: SEM-SE image of the FIB prepared sample pillar for *Mary* PXCT measurement. Scale bar: 2 µm.

S3: Supporting information regarding the Sedrun *Nicolaus* and *Bishop*

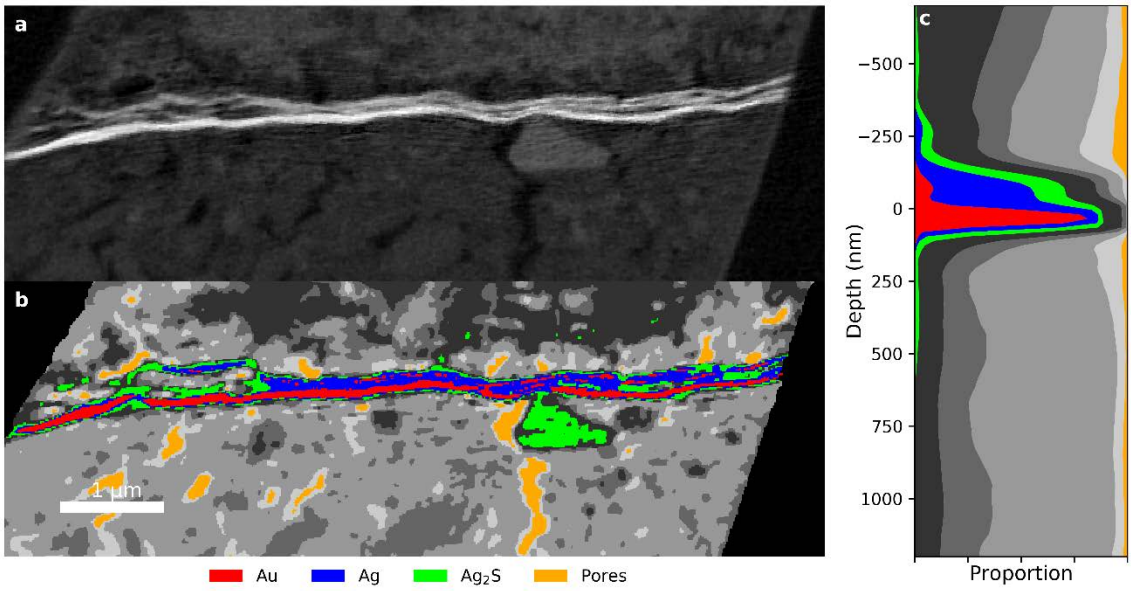

Fig. S 8: Slice through the *Nicolaus border* PXCT tomogram in (a) δ-values and (b) segmented according to the ranges shown in Fig. 6 in the main text. (c) Depth profile of the sample, aligned to the main gold-rich layer.

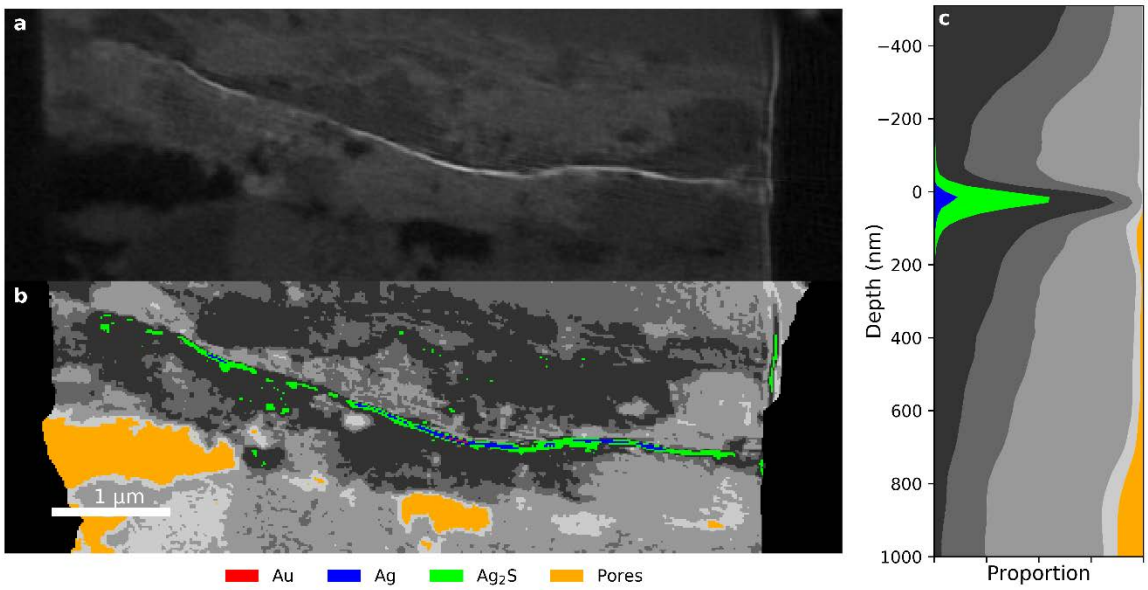

Fig. S 9: Slice through the *Nicolaus* PXCT tomogram in (a) δ-values and (b) segmented according to the ranges shown in Fig. 6 in the main text. (c) Depth profile of the sample, aligned to the metal layer.

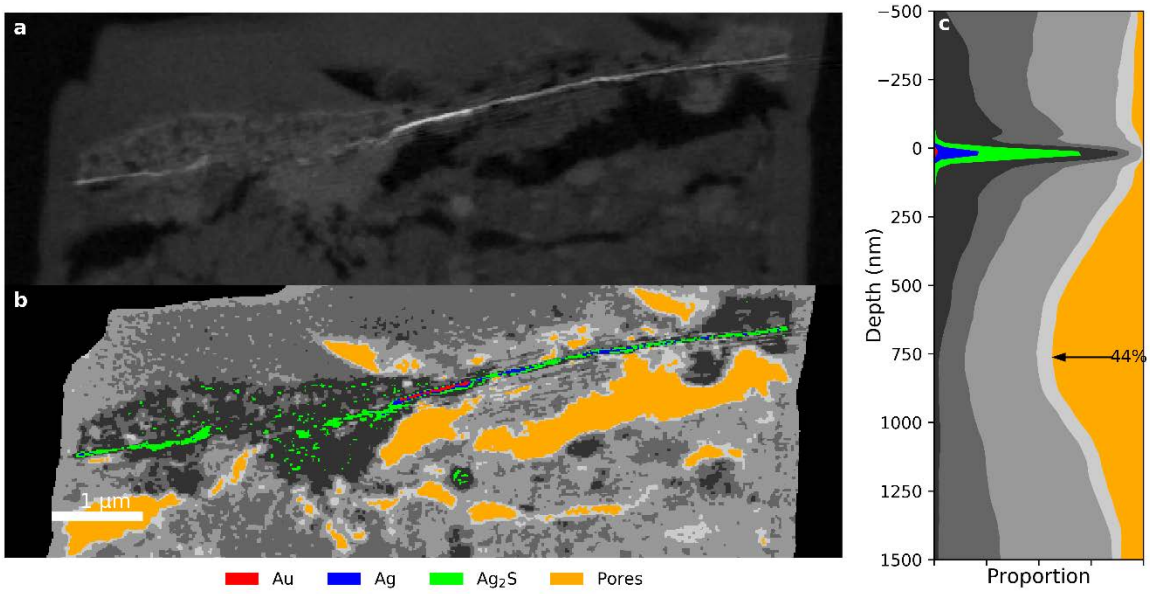

Fig. S 10: Slice through the *Bishop* PXCT tomogram in (a) δ-values and (b) segmented according to the ranges shown in Fig. 6 in the main text. (c) Depth profile of the sample, aligned to the metal layer.

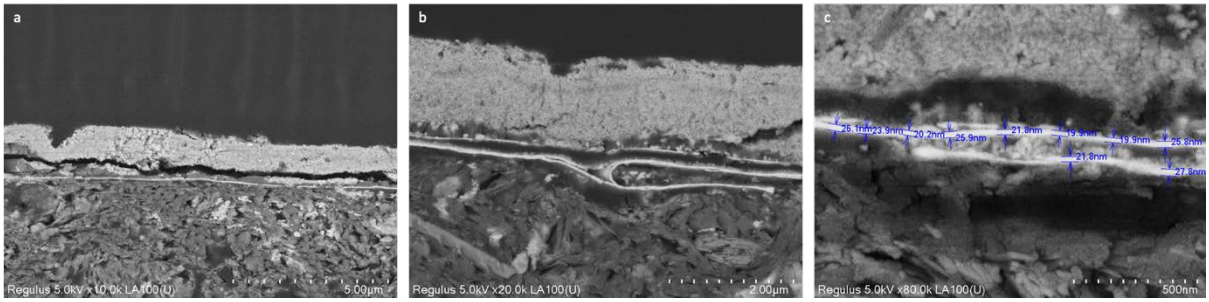

Fig. S 11: SEM-BSE images of a sample cross-section taken in an adjacent region of the *Nicolaus* sample: (a) an observation site containing corroded Zwischgold with single Au layer (Mag. 10kx); (b) an observation site containing corroded Zwischgold with multiple Au layers (Mag. 20kx); (c) thickness measurements for the Au layers (Mag. 80kx).

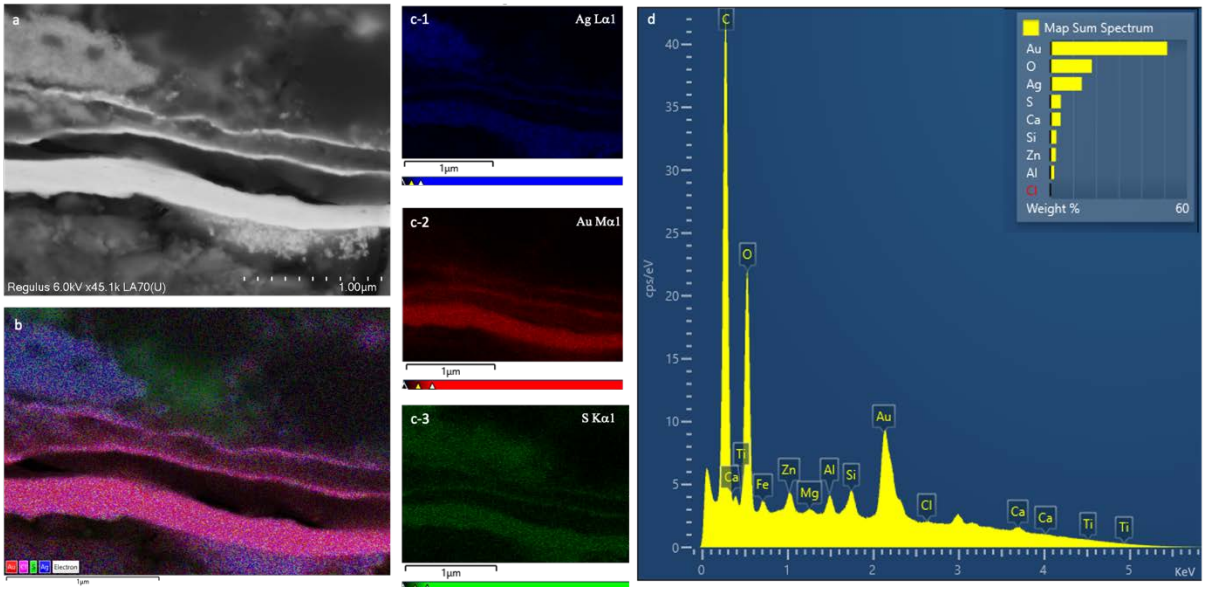

Fig. S 12: SEM-EDX measurements of a sample cross-section taken in an adjacent region to the *Nicolaus Border* sample: (a) SEM-BSE image; (b) overlay of BSE image and EDX elemental maps; (c1-3) EDX elemental maps for Ag, Au and S; (d) EDX sum spectrum of the observation site. Note that S map shows some cross-talk from the Au map, due to overlap of the Au M and S K fluorescence peaks.

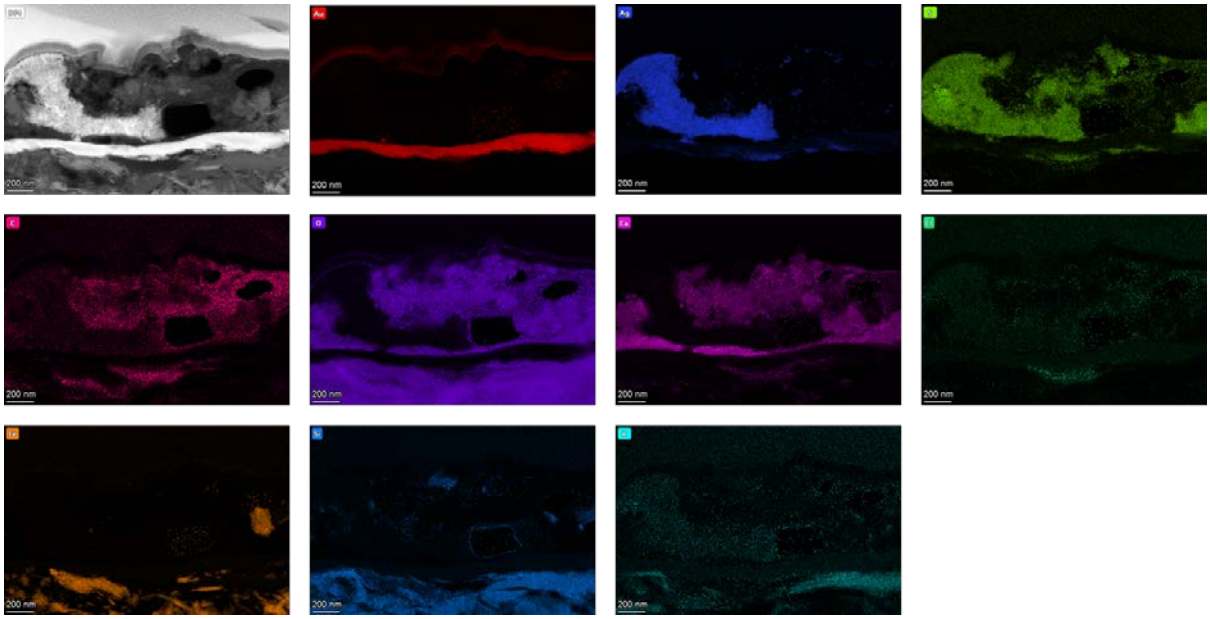

Fig. S 13: STEM image and STEM-EDX element maps of a cross-section TEM-lamella taken in a region adjacent to the *Nicolaus* sample.

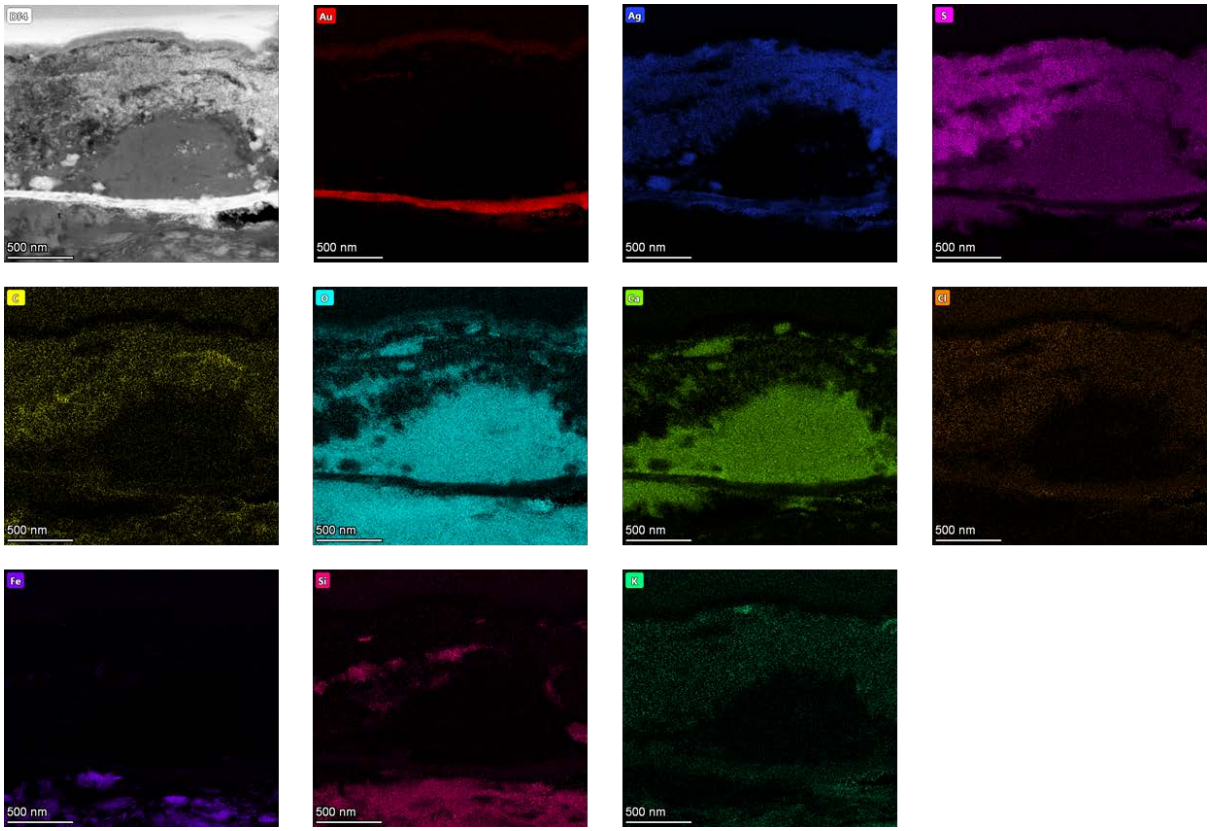

Fig. S 14: STEM image and STEM-EDX element maps of a cross-section TEM-lamella taken in a region adjacent to the *Bishop* sample.

Table S 1: Calculated δ and β values at photon energy of 8.7 keV for materials of interest in Zwischgold samples.

Material	**Type**	**Mass density (g/cm^3)**	**Number density ($x10^{22}$)**	**δ ($x10^{-6}$)**	**β ($x10^{-6}$)**
Au	Metal	19.3	5.90	**40.1**	3.74
Ag	Metal	10.49	5.86	**25.2**	2.04
Ag_2S	Corrosion product	7.2	1.75	**17.6**	1.29
AgCl	Corrosion product	5.5	2.31	**13.6**	0.934
Fe_2O_3	Bole component	5.24	1.98	**13.5**	1.06
$ZnSO_4$	Paint siccative	3.54	1.32	**9.19**	0.145
$ZnCO_3$	Paint siccative	3.5	1.68	**8.90**	0.108
$Mg_3Si_4O_{10}(OH)_2$	Bole component	2.75	4.39	**7.46**	0.004
SiO_2	Bole component	2.65	2.66	**7.13**	0.624
$Al_2Si_2O_5(OH)_4$	Bole component	2.6	0.616	**7.05**	0.322
$CaCO_3$	Bole component	2.71	1.36	**6.83**	2.85
FeO(OH)	Bole component	3.8	1.79	**6.77**	4.84
Na_2SiO_3	Bole component	2.4	1.18	**6.45**	0.304
$(CH_2)_x$	Oil/wax binder	0.9	3.86	**2.81**	0.00281
Ga	FIB ion source	5.91	5.10	**13.7**	0.332
Al	Conductive coating	2.7	6.02	**7.23**	0.114
Amorphous C	FIB protection layer	2	10.03	**5.50**	0.0076

S4: Supporting XPS depth profile measurements

The elemental composition of two samples were measured by X-ray photoelectron spectroscopy (XPS) as a function of sputter depth. The samples included a Zwischgold leaf taken from the same package as the *Recent* PXCT sample and a region adjacent to the *35-year* PXCT sample.
The measurements were performed with a PHI Quantera II (Physical Electronics) spectrometer and a micro-focused, monochromated Al K_α source (photon energy of 1486.74 eV). Access to the instrument was granted by Helmholtz-Institut Erlangen-Nürnberg (HI ERN). Sputtering was performed with Ar accelerated with 500 V onto an area of 1x1 μm^2, which corresponds to a sputter rate of about 10 nm per minute (manufacturer's calibration). Sputtering was performed for 20 seconds (about 3nm depth) per sputter cycle for the first 12 cycles and then 4 minutes (about 40 nm) per cycle for 21 sputter cycles. XPS spectra were measured after each cycle according to the parameters listed in Table S 2.

Table S 2: Parameters used for XPS measurements

Peak Attribution	Binding Energy Region (eV)	Step Size (eV)	Pass Energy (eV)	Dwell Time (ms)
Au 4f	80 – 88	0.1	13	600
Ag 3d	363 – 375	0.1	13	600
S 2p	156 – 162.5	0.1	13	600
C 1s	280 – 290	0.25	140	400
O 1s	526 – 536	0.25	140	400
I 3d5	614 – 620	0.25	140	400
Cl 2p	193 – 201	0.25	140	400

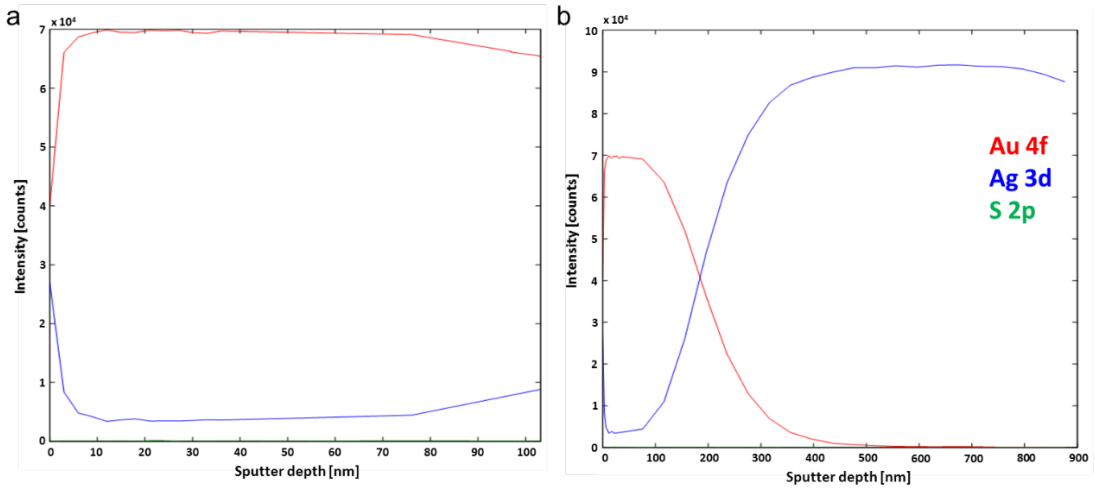

Fig. S 15: XPS depth profile of a Zwischgold leaf taken from the same package as the *Recent* sample. The intensity of the Au 4f, Ag 3d, and S 2p photoelectron peaks were tracked as a function of sputter depth. a) The first 12 sputter-measure cycles show detail in the first 100 nm below the surface. b) The full measurement set.

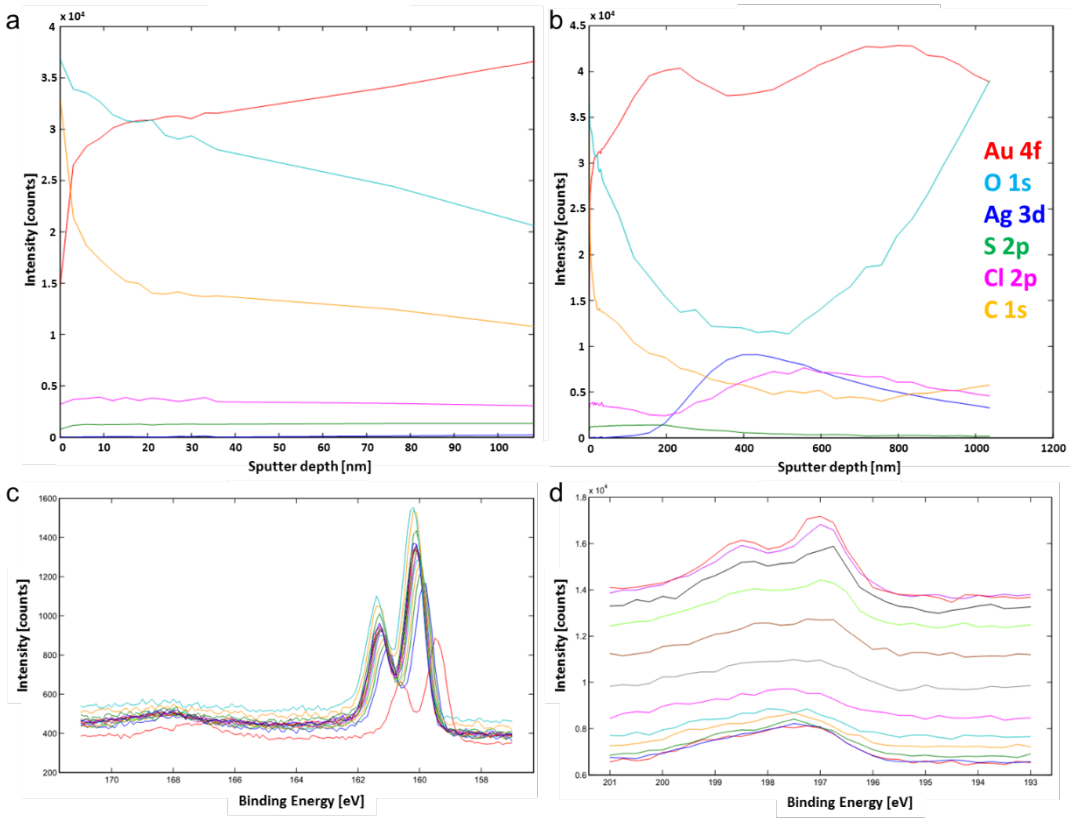

Fig. S 16: (a-b) XPS depth profile of a region adjacent to the *35-year* sample. (c) Sulfur 2p peaks from cycles 1-16 indicating presence of a sulfide compound (no signal in sulfate region). Observed peak shifts are due to charging of the sample. (d) Chlorine 2p peaks from cycles 12-23.

S5: Details of the depth profile analysis

The PXCT tomograms of the Zwischgold samples show foils with significant curvature and holes. Firstly, segmented tomograms were rotated in Avizo so that the foils were roughly aligned in the XY plane. A Python script was then run on the data via Avizo to define the reference plane position, shift the data columns to align this reference plane, and finally to sum the contributions from each segment

in each layer of the shifted tomogram. Exact details of the calculation can be read from the FoilDepthHistogram.pyscro code provided in the following section.
The position of the reference plane was calculated by examining each column of pixels in turn and finding the middle pixel of the Au segment (or the Au+Ag segment in cases of insufficient Au present) in that column. Linear interpolation was used to fill in gaps and the surface plane was smoothed by a Gaussian filter with a sigma of three pixels. The reference plane was later shifted from the middle of the segment layer to the upper surface by a vertical shift equal to half of the FWHM of the Au (or Au+Ag) segment in the depth profile.

FoilDepthHistogram.pyscro

```
import numpy, scipy, scipy.interpolate, scipy.ndimage.filters, os.path

class FoilDepthHistogram(PyScriptObject):
 def __init__(self):
  #self.data.visible = True
  self.ports.data.valid_types = ['HxUniformLabelField3']
  self.port_dest = HxConnection(self, "portDestination", "Destination")
  self.port_dest.valid_types = ['HxUniformLabelField3']
  self.do_it = HxPortDoIt(self, 'doIt', 'Apply')  # get handle to the Apply button
  self.do_it.buttons[0].enabled=False             # disable it until ready
  self.plane = HxPortGeneric(self,'plane_list', 'Plane Materials')
 def update(self):
  if not (self.data.source() is None):
   data_range = int(self.ports.data.source().range[1])
   items_list = []
   if len(self.plane.items)==0:
    for x in range(data_range+1):
     items_list.append(HxPortGeneric.GenericCheckBox(caption=str(x),checked=x in [1,2,3,4]))
    self.plane.items = items_list
   if any([self.plane.items[x].checked for x in range(data_range+1)]):
    self.do_it.buttons[0].enabled=True
   else:
    self.do_it.buttons[0].enabled=False
  else:
   self.do_it.buttons[0].enabled=False

  PyScriptObject.update(self)

 def __plot_histogram(self):
  if self.__corr_plot == None :
   self.__corr_plot = hx_project.create('HxCorrelationPlot')
  self.__corr_plot.ports.source1.connect(self.ports.data.source())
  self.__corr_plot.ports.source2.connect(self.__grad_mag)
  self.__corr_plot.ports.action.buttons[0].hit = True
  self.__corr_plot.fire()

  self.viewerPlot = hx_project.create('HxPlot2Viewer')
  self.viewerPlot.ports.PlotModule.connect(self.__corr_plot)
  self.viewerPlot.viewer_mask = 65520
  self.viewerPlot.ports.actions.toggles[1].checked = HxPortToggleList.Toggle.CHECKED
  self.viewerPlot.fire()


 def MakeFoilPlane(self, data, plane_material,guide_plane=None,width=numpy.infty):
  if guide_plane is None:
   guide_plane = numpy.full(data.shape[:2],data.shape[2]/2)
  width = max(width,1)
  foil_plane = numpy.full(data.shape[:2],numpy.nan)
  I=[]
  J=[]
  V=[]
  for x in range(data.shape[0]):
   for y in range(data.shape[1]):
    line = max(0,int(guide_plane[x,y])-width)
+numpy.where(numpy.in1d(data[x,y,max(0,int(guide_plane[x,y])-width)
:min(data.shape[2],int(guide_plane[x,y])+width)].flatten(),plane_material))[0]
min(data.shape[2],int(guide_plane[x,y])+width)
    if len(line) > 0:
     foil_plane[x,y] = numpy.median(line)
     I.append(x)
```

```
      J.append(y)
      V.append(numpy.median(line))

  I = numpy.array(I)
  J = numpy.array(J)
  V = numpy.array(V)
  grid_y, grid_x =
numpy.meshgrid(numpy.arange(foil_plane.shape[1]),numpy.arange(foil_plane.shape[0]))
  edges = numpy.ones(foil_plane.shape)
  edges[1:-1,1:-1] = 0
  edgesi = numpy.where(edges)
  foil_plane[edgesi] = scipy.interpolate.griddata(zip(I,J), V, edgesi, method='nearest')
  bad_points = ~numpy.isfinite(foil_plane)
  foil_plane[bad_points] = scipy.interpolate.griddata(zip(grid_x[~bad_points],
grid_y[~bad_points]), foil_plane[~bad_points], (grid_x[bad_points], grid_y[bad_points]),
method='linear')
  return foil_plane

 def RefinePlane(self, foil_plane,sigma):
  print "sigma is: ", sigma
  smooth_plane = scipy.ndimage.filters.gaussian_filter(foil_plane, sigma=sigma)
  diff_plane = numpy.abs(foil_plane-smooth_plane)
  outliers = diff_plane>2*numpy.median(diff_plane)
  print "diff is: ", numpy.median(diff_plane), "\t(",
100.0*numpy.sum(outliers)/float(foil_plane.shape[0]*foil_plane.shape[1]), "%)"
  edges = numpy.ones(foil_plane.shape)
  edges[1:-1,1:-1] = 0
  edgesi = numpy.where(edges)
  outliers = numpy.logical_or(outliers,edges==1)
  foil_plane2 = foil_plane.copy()
  grid_y, grid_x =
numpy.meshgrid(numpy.arange(foil_plane.shape[1]),numpy.arange(foil_plane.shape[0]))
  foil_plane2[edgesi] = scipy.interpolate.griddata(zip(grid_x[~outliers], grid_y[~outliers]),
foil_plane2[~outliers], edgesi, method='nearest')
  outliers = numpy.logical_and(outliers,edges==0)
  foil_plane2[outliers] = scipy.interpolate.griddata(zip(grid_x[~outliers],
grid_y[~outliers]), foil_plane2[~outliers], (grid_x[outliers], grid_y[outliers]),
method='linear')
  bad_points = ~numpy.isfinite(foil_plane2)
  if numpy.sum(bad_points)>0:
   foil_plane2[bad_points] = scipy.interpolate.griddata(zip(grid_x[~bad_points],
grid_y[~bad_points]), foil_plane2[~bad_points], (grid_x[bad_points], grid_y[bad_points]),
method='nearest')
  return foil_plane2, numpy.median(diff_plane)

 def compute(self):
  if not self.do_it.was_hit:
   return
  print "compute!"
  data = self.data.source().get_array().copy()
  plane_material = numpy.where([self.plane.items[x].checked for x in
range(int(self.ports.data.source().range[1])+1)])[0]

  foil_plane = self.MakeFoilPlane(data, plane_material)
  [foil_plane, diff] = self.RefinePlane(foil_plane,sigma=0.2*numpy.min(foil_plane.shape))
  foil_plane = self.MakeFoilPlane(data, plane_material, foil_plane, width=int(diff))
  [foil_plane, diff] = self.RefinePlane(foil_plane,sigma=3)
  foil_plane = foil_plane.astype(int)

  padded_data = numpy.pad(data,((0,0),(0,0),(data.shape[2],0)),'constant')
  print padded_data.shape, padded_data[1,1,:].shape
  for x in range(data.shape[0]):
   for y in range(data.shape[1]):
    padded_data[x,y,:] = numpy.roll(padded_data[x,y,:],-foil_plane[x,y])
  padded_data = padded_data[:,:,data.shape[2]-numpy.amax(foil_plane):-numpy.amin(foil_plane)]
  plane_height = padded_data.shape[2]-numpy.amax(foil_plane)
  del foil_plane

  DepthProfile = numpy.zeros((numpy.amax(data)+1,padded_data.shape[2]))
  bin_set = list(numpy.arange(-1,numpy.amax(data)+1)+0.5)
  for z in range(padded_data.shape[2]):
   DepthProfile[:,z] = numpy.histogram(padded_data[:,:,z],bins=bin_set)[0]

  source = self.data.source()
  output = source.duplicate()
  [s_base, s_ext] = os.path.splitext(source.name)
  if s_ext == '.am':
```

```
   output.name = s_base+".rolled"
  else:
   output.name = source.name+".rolled"
  histogram_filename = "L:\Analysis\%s_histogram.txt" % output.name
  output.set_array(padded_data)
  BB = numpy.array(source.bounding_box)
  print BB
  BB[1,2] = (BB[1,2]-BB[0,2])/(data.shape[2]-1)*(padded_data.shape[2]-1) + BB[0,2]
  print BB
  output.bounding_box = BB
  DepthProfileRange = numpy.linspace(0,BB[1,2]-BB[0,2],num=padded_data.shape[2])
  DepthProfileRange = DepthProfileRange-DepthProfileRange[plane_height]
  hx_project.add(output)
  print 'Pixel size is ', (numpy.array(source.bounding_box[1]) -
numpy.array(source.bounding_box[0]))/numpy.subtract(data.shape,1)

  header = 'Depth\t'
  for i in range(DepthProfile.shape[0]):
   header += 'Material %i\t'%i
  DepthProfile = numpy.vstack((DepthProfileRange,DepthProfile))

numpy.savetxt(histogram_filename,DepthProfile.T,fmt='%1.3f\t'+'%i\t'*(DepthProfile.shape[0]-
1),header=header)
  print 'Saved histogram to  ', histogram_filename

  PyScriptObject.compute(self)
```

S6: FIB-sectioning

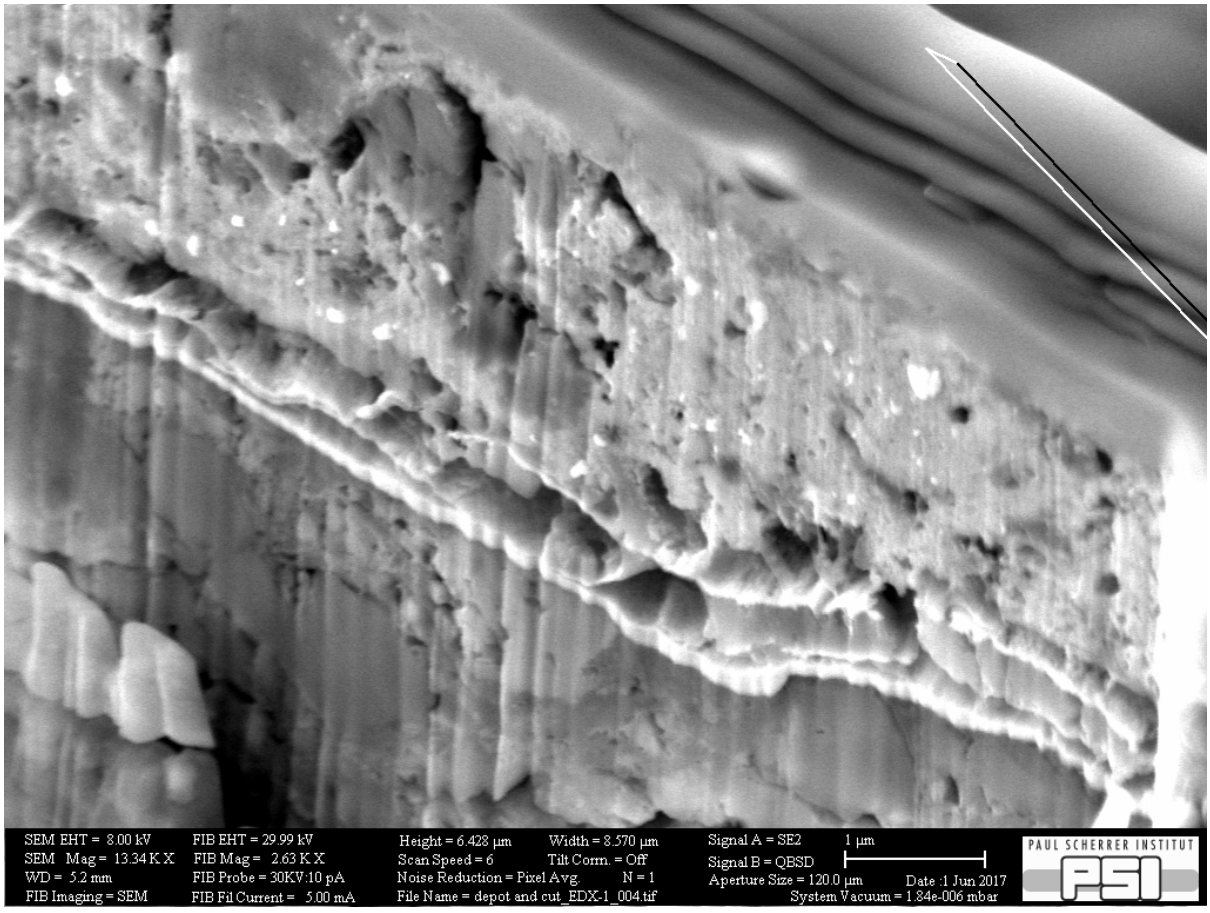

Fig. S 17: FIB-sectioning of the *Nicolaus Border* sample resulted in strong curtaining effects due to strong charging and heterogeneity (varying sputter rates) of the sample materials.

S7: Why discard the Beta part of the data?

The ptychographic imaging used in PXCT produces both the absorption (Beta) and dispersion (Delta) parts of the sample material's refractive index. However, the Beta part of the data was discarded in this work, as is often the case with hard X-ray PXCT measurements. Figure S18 demonstrates why this is the case. The four material peaks are clearly much broader in the Beta axis and much better separated along the Delta axis. Note that while the Delta axis spans a range more than four times that of the Beta axis in Figure S18, the line running through the peaks is close to vertical and thus indicates that almost all of the contrast is contained within the Delta channel. This pattern is typical for hard X-ray PXCT measurements because absorption effects are usually very weak and both Delta and Beta tend to both scale roughly proportional to the electron density of the material. Exceptions to this rule can occur when the photon energy is close to a transition resonance (not the case in the measurements presented in this work) where the observed Beta and Delta values can differ significantly from the non-resonant values.

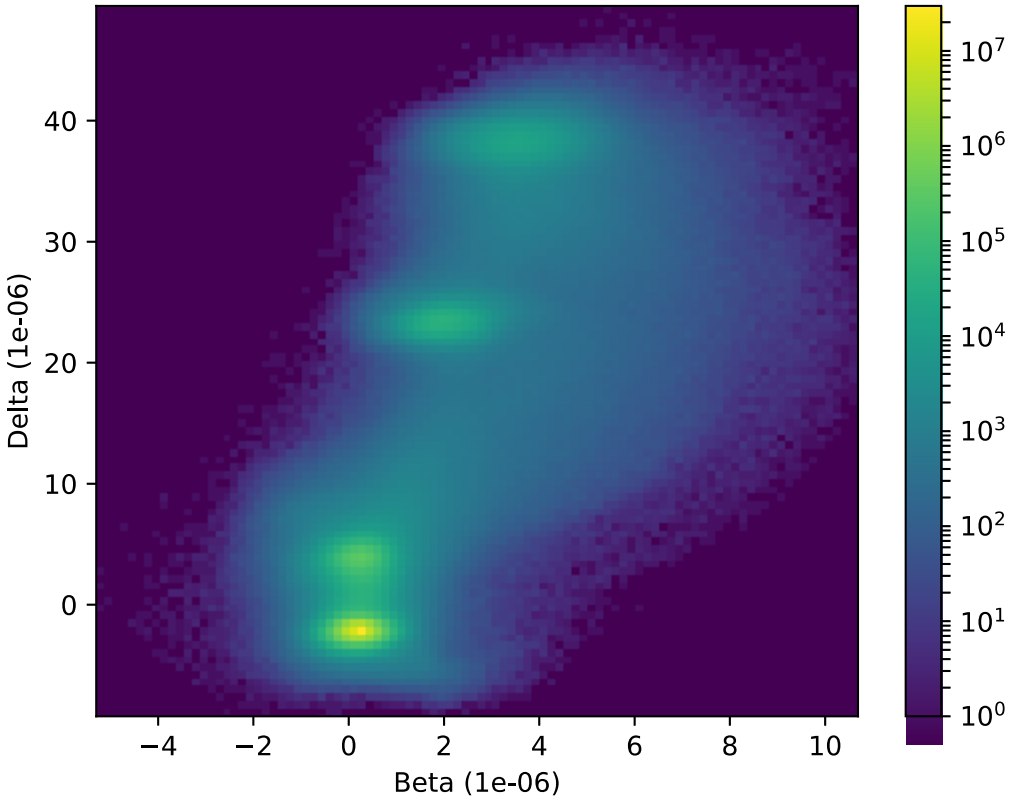

Fig. S 18 Bivariate histogram of tomogram voxels for the *10-year* PXCT sample. Each pixel in this image indicates the number of voxels from the tomogram having Beta and Delta values corresponding to the position of the pixel on the plot axes. Summing all pixels along each row gives the Delta histogram shown in Figure 3 of the main text (green line), while summing along each column would give a Beta histogram. The four peaks correspond to (from top to bottom) gold, silver, carbon and air.