Attachment 1 to: Leuchs AK, Neuhäuser M. A SAS/IML algorithm for exact nonparametric paired tests. GMS Med Inform Biom Epidemiol. 2010;6(1):Doc04. DOI: 10.3205/mibe000104, URN: urn:nbn:de:0183-mibe0001041
Freely available from: http://www.egms.de/en/journals/mibe/2010-6/mibe000104.shtml

Appendix

```
/*
*data:            data set
*label_diff:      name of variable (difference) to evaluate
*test:            Specification of the test
                              - 'pratt' for test according to Pratt
                              - 'signed' for Wilcoxon's signed rank test
                              - 'original' for test based on original data
*round:                 specifies the number of decimal places to round on
                        (rounding only for the test based on original data)
                        optional parameter: DEFAULT=4
*/

%MACRO signedrank(data, label_diff, test, round=4);

data data;
set &data(rename=(&label_diff=obs_diff));
diff_abs=abs(obs_diff);
keep diff_abs obs_diff;
run;

proc sql noprint;
select count(*) into :diff0 from data where diff_abs=0;
select count(*) into :n_all from data;
quit;

*abort if all differences are zero;
%IF &n_all=&diff0 %Then %DO;
      %put NOTE: all differences are zero;
      %RETURN;
%END;

*assigning ranks for nonzero values;
proc rank data=data out=data;
where obs_diff<>0;
var diff_abs;
ranks rank_diff;
run;

*signed ranks;
data data;
set data;
rank_sign=rank_diff*sign(obs_diff);
run;

proc iml;
use data;
*Wilcoxons signed rank test;
if &test='signed' then do;
      read all var {rank_sign} into d;    *signed ranks;
end;
*Pratts modification of Wilcoxons signed rank test;
if &test='pratt' then do;
      read all var {rank_sign} into  d; *signed ranks;
      d=(abs(d)+&diff0)#sign(d);    *assigning  ranks  according  to  Pratts
modification;
end;
*test based on original data;
if &test='original' then do;
      read all var {obs_diff} into d;
      *rounding;
      d=round((10**&round)*d);
end;
*computation of the statistic;
tstat = sum(d#(d>=0));

*shift-algorithm;
start shift(d);
      n=NROW(d);
      *for the shift-algorithm the values in d need to be integer;
      potenz=0;
      do while (sum(d^=int(d))^=0);
            d=10*d;
            potenz=potenz+1;
      end;
      ad = abs(d); *absolute values;
      *determine largest common factor;
      ggt=0;
      k = min(ad + (d=0)#max(ad)); *smallest |d| ^= 0;
      do while (ggt = 0 & k>=1);
            if d/k = int(d/k) then ggt=k;
            else k=k-1;
      end;
      *required constants;
      adshort = ad/ggt;
      lng = sum(adshort);
      values = ((0:lng)*ggt)`; *possible values for statistic;
      *algorithm;
      if lng <> 0 then do;
            prob = 1 // j(lng,1,0);
            do k=1 to n;
                  if adshort[k] <> 0 then do;
                        shift = j(adshort[k],1,0) // prob[1:lng+1-
                                adshort[k]];
                        prob = prob + shift;
                  end;
                  else prob = 2*prob; *difference can be 0;
            end;
            prob = prob/(2**n);
            *reverse the process which changes the observations/ranks
            to integers (this was needed for the shift algorithm);
            values=values/(10**potenz);
            *resulting distribution (1th column: statistic -- 2nd column:
             probability);
            dist = values || prob;
      end;
      d=d*10**(-1*potenz);
      return (dist);
 finish;

 dist=shift(d);

 *computation of p-values;
 *one-sided;
 pvalue_gr=sum(dist[,2]#(dist[,1]>=tstat));
 pvalue_less=sum(dist[,2]#(dist[,1]<=tstat));

 *two-sided;
 tstatOUT=tstat; *output-statistic;
 if (&test='signed') then do;
      ew=(&n_all-&diff0)*((&n_all-&diff0)+1)/4;
 end;
 else if (&test='pratt') then do;
      ew=((&n_all*(&n_all+1)-(&diff0*(&diff0+1)))/4);
 end;
 else if (&test='original') then do;
      tstatOUT=tstat/(10**&round);
      ew=sum(abs(d))/2;
 end;
 if (tstat<ew) then do;

      d=-1*d;
      tstat=sum(d#(d>=0));
 end;
 lower=ew-(tstat-ew);
 pvalue_two=sum(dist[,2]#(dist[,1]<=lower | dist[,1]>=tstat));

 *OUTPUT;
 label={'n'     'n_nonzero'     'statistic'     'pvalue_gr'     'pvalue_less'
'pvalue_two'};
 out= &n_all || &n_all-&diff0 || tstatOUT || pvalue_gr || pvalue_less ||
      pvalue_two;

 create output from out [colname=label];
 append from out;
 quit;

 data output; set output; test=&test; run;

 proc report data=output nowd headline;
 column test n n_nonzero statistic pvalue_less pvalue_gr pvalue_two;
 define n / display 'n';
 define n_nonzero   / display 'n (nonzero)';
 define statistic   / display 'statistic';
 define pvalue_less / display 'p-value (less)';
 define pvalue_gr   / display 'p-value (greater)';
 define pvalue_two  / display 'p-value (two-sided)';
 run;

%MEND;
```