```c
#include <math.h>
#include <stdlib.h>
#include "declar.h"

/* Computes the Lomb-Scargle normalised periodogram of a times-series.
 *
 * t the times, reduced to [-1/2,1/2).
 * y the measurements, centred around <y>.
 * npts the length of the times-series.
 * over the oversampling factor.
 * hifac the highest frequency in units of "average" Nyquist frequency.
 *
 * This function returns the results in a structure, LS (see text).
 */
LS* periodogram(const double* t, const double* y, int npts, double over, double hifac)
{
  double df = 1.0 / (over * (t[npts - 1] - t[0]));

  // Index of the highest frequency in the positive frequency part of spectrum.
  int m = floor(0.5 * npts * over * hifac);
  LS* ls = xmalloc(sizeof(LS));
  ls->freqs = xmalloc(m * sizeof(double));
  ls->Pn = xmalloc(m * sizeof(double));
  ls->nfreqs = m;

  // Unnormalised FTs of the data and window.
  double complex* sp = xmalloc((m + 1) * sizeof(double complex);
  nfft(t, y, npts, m, sp);
  double complex* win = xmalloc((2 * m + 1) * sizeof(double complex);
  nfft(t, NULL, npts, 2 * m, win);

  // Computes the periodogram ordinates,
  // and store the results in the LS structure.
  for (int j = 1; j <= m; j++)
  {
    double complex z1 = sp[j];      // FT of data at \omega
    double complex z2 = win[2 * j]; // FT of window at 2\omega
    double absz2 = cabs(z2);
    double hc2wt = 0.5 * cimag(z2) / absz2;
    double hs2wt = 0.5 * creal(z2) / absz2;
    double cwt = sqrt(0.5 + hc2wt);
    double swt = sign(sqrt(0.5 - hc2wt), hs2wt);
    double den = 0.5 * npts + hc2wt * creal(z2) + hs2wt * cimag(z2);
    double cterm = square(cwt * creal(z1) + swt * cimag(z1)) / den;
    double sterm = square(cwt * cimag(z1) - swt * creal(z1)) / (npts - den);
    ls->freqs[j - 1] = (j - 1) * df;
    ls->Pn[j - 1] = (cterm + sterm) / (2 * var);
  }

  free(win);
  free(sp);
  return ls;
}
```