```
//
// Programmer:    Craig Stuart Sapp <craig@ccrma.stanford.edu>
// Creation Date: Sat May 13 12:16:45 PDT 2006
// Last Modified: Sat May 20 15:50:06 PDT 2006 (parameters control added)
// Last Modified: Sun May  6 01:48:58 PDT 2007 (upgraded to vamp 1.0)
// Filename:      MzPowerCurve.h
// URL:           http://sv.mazurka.org.uk/include/MzPowerCurve.h
// Documentation: http://sv.mazurka.org.uk/MzPowerCurve
// Syntax:        ANSI99 C++; vamp 1.0 plugin
//
// Description:   Calculate the power of an audio signal as it changes
//                over time.
//

#ifndef _MZPOWERCURVE_H_INCLUDED
#define _MZPOWERCURVE_H_INCLUDED

#include "MazurkaPlugin.h"  // Mazurka plugin interface for Sonic Visualiser
#include "MazurkaWindower.h"

#include <list>


class MzPowerCurve : public MazurkaPlugin {

   public:

   // plugin interface functions:

                    MzPowerCurve            (float samplerate);
      virtual      ~MzPowerCurve            ();

      // required polymorphic functions inherited from PluginBase:
      std::string   getIdentifier           (void) const;
      std::string   getName                 (void) const;
      std::string   getDescription          (void) const;
      std::string   getMaker                (void) const;
      std::string   getCopyright            (void) const;
      int           getPluginVersion        (void) const;

      // optional parameter interface functions
      ParameterList getParameterDescriptors (void) const;

      // required polymorphic functions inherited from Plugin:
      InputDomain   getInputDomain          (void) const;
      OutputList    getOutputDescriptors    (void) const;
      bool          initialise              (size_t channels,
                                             size_t stepsize,
                                             size_t blocksize);
      FeatureSet    process                 (AUDIODATA inputbufs,
                                             Vamp::RealTime timestamp);
      FeatureSet    getRemainingFeatures    (void);
      void          reset                   (void);

      // optional polymorphic functions from Plugin:
      size_t        getPreferredStepSize    (void) const;
      size_t        getPreferredBlockSize   (void) const;
      // size_t     getMinChannelCount      (void) const { return 1; }
      // size_t     getMaxChannelCount      (void) const { return 1; }

   // non-interface functions and variables:

      static double getStandardDeviation    (std::vector<double>& data);
      static double getMean                 (std::vector<double>& data);

   private:

      int mz_filterforward;         // true if forward filtering
      int mz_filterbackward;        // true if reverse filtering

      MazurkaWindower mz_window;    // used for weighted averaging
      double          mz_windowsum; // for normalization of weighted power
      std::vector<double> rawpower; // power data for non-causal calculations

   // plugin parameters:
   //    "windowsize"         -- size of the analysis window in milliseconds
   //    "hopsize"            -- distance between window start times in ms
   //    "smoothingfactor"    -- gain value for exponential smoothing filter
   //    "filtermethod"       -- which way to filter raw power
   //    "cutoffthreshold"    -- noise floor in dB
   //    "cutoffwidth"        -- transition region around threshold in dB

};


#endif // _MZPOWERCURVE_H_INCLUDED
```