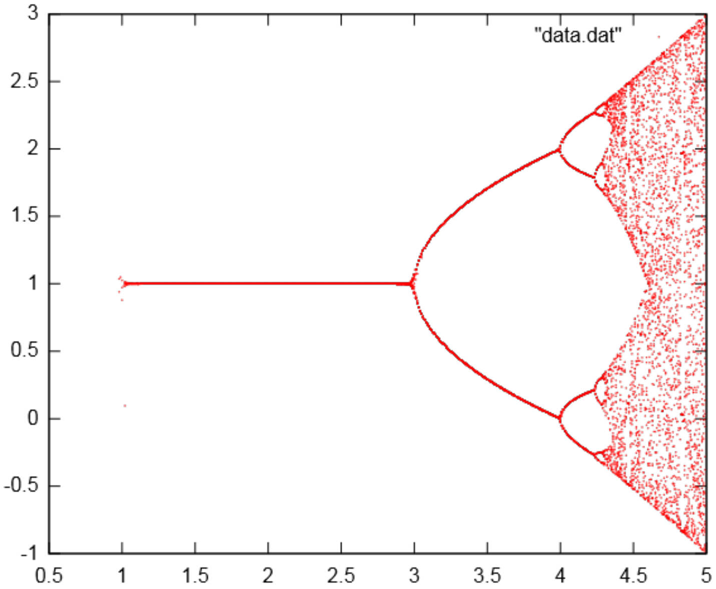
Source Code

```c
#include "stdio.h"
#include "stdlib.h"
#include "math.h"
#include "string.h"

void plot( char *input_file, char *output_file )
{
  FILE *gnuplot;
  gnuplot = popen("gnuplot", "w");
  if( output_file )
  {
    fprintf(gnuplot, "set term svg\n");
    fprintf(gnuplot, "set out \"%s\"\n", output_file );
  }
  fprintf(gnuplot, "plot \"%s\" with dots\n", input_file);
  fflush(gnuplot);
  fclose(gnuplot);
}

double example_function( double param, double point )
{
  return pow(point,3) - 3*pow(point,2) + (5-param)*point - 2 + param;
}

void bifurcation_diagram( int param_min, int param_max, double param_step,
             int point_min, int point_max, int num_points,
             double (*f)(double,double), int num_iter, int tolerancy)
{
  FILE* file;
  double param, point;
  int i,j;

  srand(time(NULL));
  file = fopen("data.dat","w");

  for ( param = param_min; param < param_max; param += param_step )
  {
    for ( i = 0; i < num_points; i++ )
    {
      point = point_min + ((double) rand() / (double) RAND_MAX) * (point_max-
         point_min);

      for ( j = 0; j < num_iter && abs(point) < tolerancy; j++ )
      {
        point = (*f)(param,point);
      }

      if(abs(point) < tolerancy)
      {
        fprintf(file,"%lf %lf\n",param, point);
      }
    }
  }
  plot( "data.dat", "graph.svg" );
}
```

Example

```
int main(int argc, char const *argv[])
{
  bifurcation_diagram( 0, 5, 10e-3, 0, 5, 100, &example_function, 100, 10e1);

  return 0;
}
```

Output


"data.dat"