Introduction to Computer Programming

Carolyne Pickler, Neil Edelman

January 12, 2007

Abstract

Given the data supplied for this instructional lab, several operations were performed upon it. These values represent an unknown signal.

1 Introduction

Using programming language, “C”, programmes to calculate the square, square-root, derivative, integral, and root-mean-square of the given unknown signal were written. The value of these programmes is to be able to calculate these operations on large amounts of data in a short amount of time.

2 Theory

The derivative was calculated using,

$$slope = \frac{in_y - prev_y}{in_x - prev_x} \tag{1}$$

This equation calculates the slope of the given data, which is equivalent to its derivative. The integral was calculated using

$$area = \frac{1}{2}(in_x - prev_x)(in_y - prev_y) \tag{2}$$

This is the midpoint method to calculate the area. The area was added to the previous value, resulting in a good approximation of the integal of the given the data.

3 Data

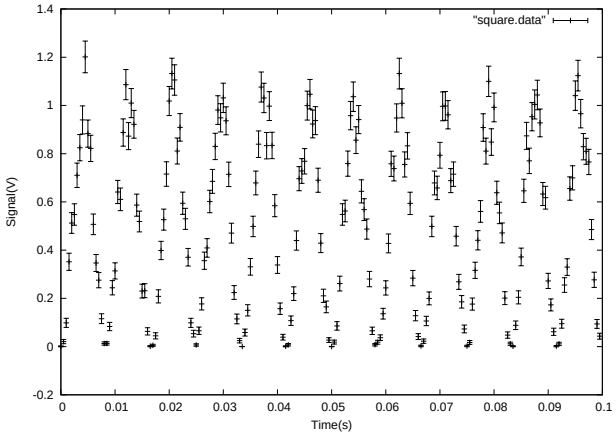

Figure 1: Square of the Signal as a Function of Time

All the values seen in Figure 1 are positive, as this is a square function.

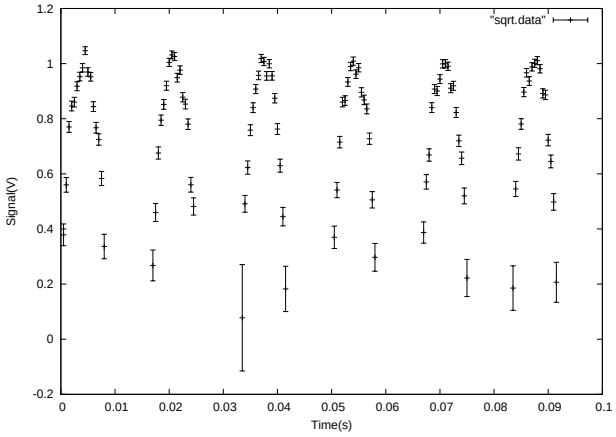

Figure 2: Square-Root of the Signal as a Function of Time

All the values seen in Figure 2 are positive, as this is a square-root function.

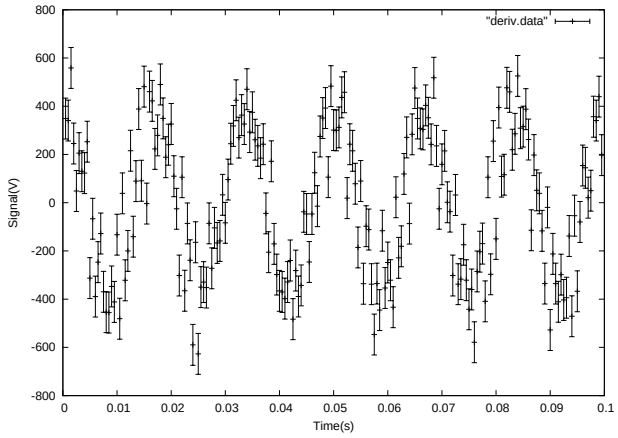

Figure 3: Derivative of the Signal as a Function of Time

The distribution of the values in Figure 3 resembles a cosine function. This is expected.

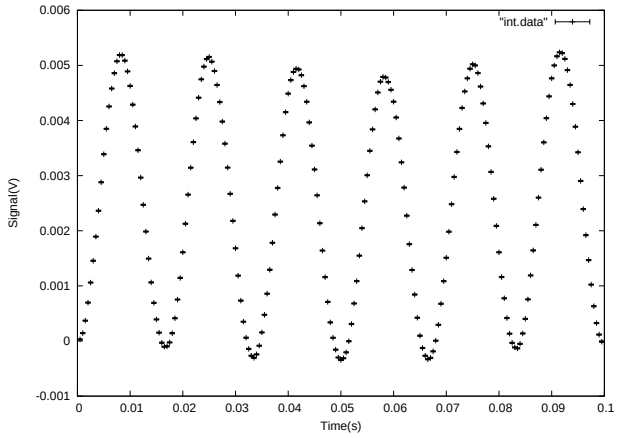

Figure 4: Integral of the Signal as a Function of Time

As expected, the distribution of the values in Figure 4 resembles a negative cosine function.

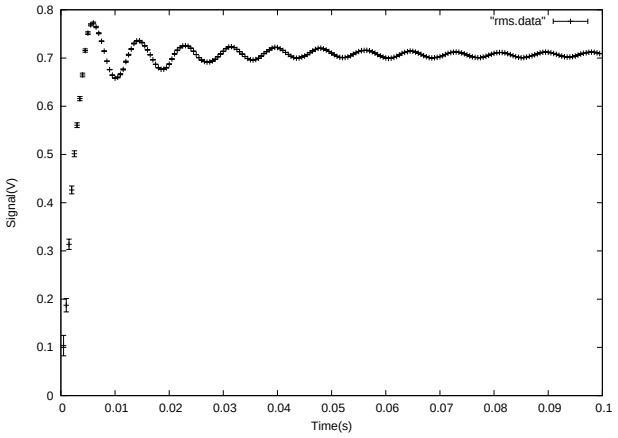

Figure 5: Root-Mean-Square of the Signal as a Function of Time

The values seen in Figure 5 are all positive, which is expected because the root-mean-square is the square root of the square of the mean of the dependent variable.

4 Conclusions

As a result of performing the operations on the unknown signal, we were able to determine that it is a sinusoidal signal. This can be seen by examining the results obtained when the derivative of the signal was taken. As expected from a sinusoidal signal, the result is a cosine. The same is true when the integral of the signal was taken. As seen in Figure 4, the result is a negative cosine wave, which coincides with what is expected from the integral of a sinusoidal function. Possible sources for this signal include anything that exhibits simple harmonic motion, such as a pendulum, or an everyday electrical outlet.

A Programme to Compute the Square of the Dependent Variable: square.c

```
#include <stdio.h>

/*
 * A very simple program to read in three columns X, Y, Z
 * and print out them back out again
 *
 */

int main (void)
{
  char line [256]; /* used to buffer a whole line of input */

  double in_x, in_y, in_e; /* input variables */
  double out_x, out_y, out_e; /* output variables */
  double prev_x, prev_y, prev_e; /* transitory variables */

  int counter = 0, rc; /* used for debugging bad data input */

  while (fgets (line, 256, stdin)) { /* read a whole line into buffer */
    counter++; /* increment line counter */
    rc = sscanf (line, "%lf %lf %lf", &in_x, &in_y, &in_e); /* attempt to parse variables */
    if (rc != 3) { /* if didn't find all three protest and exit */
      fprintf (stderr, "Less than three floats on line %i:\n\"%s\"\n", counter, line);
      return 1;
    }

    out_x = in_x; /* calculate  */
    out_y = in_y * in_y;
    out_e = 2 * in_y * in_e;

    printf ("%g\t%g\t%g\n", out_x, out_y, out_e); /* calculate and output result */

    prev_x = in_x; /* save the previous triplet */
    prev_y = in_y;
    prev_e = in_e;
  }
  fprintf (stderr, "Processed %i lines\n", counter); /* babble some statistics */
  return 0;
}
```

B Programme to Compute the Square-Root of the Dependent Variable: sqrt.c

```
#include <stdio.h>
#include <math.h>

/*
 * A very simple program to read in three columns X, Y, Z
 * and print out them back out again
 *
 */
```

```
int main (void)
{
  char line [256]; /* used to buffer a whole line of input */

  double in_x, in_y, in_e; /* input variables */
  double out_x, out_y, out_e; /* output variables */
  double prev_x, prev_y, prev_e; /* transitory variables */

  int counter = 0, rc; /* used for debugging bad data input */

  while (fgets (line, 256, stdin)) { /* read a whole line into buffer */
    counter++; /* increment line counter */
    rc = sscanf (line, "%lf %lf %lf", &in_x, &in_y, &in_e); /* attempt to parse variables */
    if (rc != 3) { /* if didn't find all three protest and exit */
      fprintf (stderr, "Less than three floats on line %i:\n\"%s\"\n", counter, line);
      return 1;
    }

    out_x = in_x; /* calculate  */
    out_y = sqrt(in_y);
    out_e = in_e / (2 * out_y);

    printf ("%g\t%g\t%g\n", out_x, out_y, out_e); /* calculate and output result */

    prev_x = in_x; /* save the previous triplet */
    prev_y = in_y;
    prev_e = in_e;
  }
  fprintf (stderr, "Processed %i lines\n", counter); /* babble some statistics */
  return 0;
}
```

C Programme to Compute the Derivative of the Dependent Variable: deriv.c

```
#include <stdio.h>
#include <math.h>

/*
 * A very simple program to read in three columns X, Y, Z
 * and print out them back out again
 *
 */

int main (void)
{
  char line [256]; /* used to buffer a whole line of input */

  double in_x, in_y, in_e; /* input variables */
  double out_x, out_y, out_e; /* output variables */
  double prev_x = 0, prev_y = 0, prev_e = 0; /* transitory variables */

  int counter = 0, rc; /* used for debugging bad data input */
```

```
  while (fgets (line, 256, stdin)) { /* read a whole line into buffer */
    counter++; /* increment line counter */
    rc = sscanf (line, "%lf %lf %lf", &in_x, &in_y, &in_e); /* attempt to parse variables */
    if (rc != 3) { /* if didn't find all three protest and exit */
      fprintf (stderr, "Less than three floats on line %i:\n\"%s\"\n", counter, line);
      return 1;
    }

    out_x = in_x; /* calculate  */
    out_y = in_y;
    out_e = in_e;

    if(counter > 1) {
      double a, b;              /* temp var */

      /* calculate the derivative */
      out_y = (in_y - prev_y) / (in_x - prev_x);
      a = in_e / (in_x - prev_x);
      b = prev_e / (prev_x - in_x);
      out_e = sqrt(a * a + b * b);

      printf ("%g\t%g\t%g\n", out_x, out_y, out_e); /* calculate and output result */
    }

    prev_x = in_x; /* save the previous triplet */
    prev_y = in_y;
    prev_e = in_e;
  }
  fprintf (stderr, "Processed %i lines\n", counter); /* babble some statistics */
  return 0;
}
```

D Programme to Compute the Integral of the Dependent Variable: int.c

```
#include <stdio.h>
#include <math.h>

/*
 * A very simple program to read in three columns X, Y, Z
 * and print out them back out again
 *
 */

int main (void)
{
  char line [256]; /* used to buffer a whole line of input */

  double in_x, in_y, in_e; /* input variables */
  double out_x, out_y, out_e; /* output variables */
  double prev_x = 0, prev_y = 0, prev_e = 0; /* transitory variables */
  double integral = 0;          /* used to calculate the running int(0, sin(x)) */

  int counter = 0, rc; /* used for debugging bad data input */
```

```
  while (fgets (line, 256, stdin)) { /* read a whole line into buffer */
    counter++; /* increment line counter */
    rc = sscanf (line, "%lf %lf %lf", &in_x, &in_y, &in_e); /* attempt to parse variables */
    if (rc != 3) { /* if didn't find all three protest and exit */
      fprintf (stderr, "Less than three floats on line %i:\n\"%s\"\n", counter, line);
      return 1;
    }

    out_x = in_x; /* calculate  */
    out_y = in_y;
    out_e = in_e;

    if(counter > 1) {
      double a, b;              /* temp var */

      /* calculate the derivative */
      out_y = (in_x - prev_x) * (in_y + prev_y) / 2;
      integral += out_y;
      a = (in_x - prev_x) * in_e / 2;
      b = (-prev_x + in_x) * prev_e / 2;
      out_e = sqrt(a * a + b * b);

      printf ("%g\t%g\t%g\n", out_x, integral, out_e); /* calculate and output result */
    }

    prev_x = in_x; /* save the previous triplet */
    prev_y = in_y;
    prev_e = in_e;
  }
  fprintf (stderr, "Processed %i lines\n", counter); /* babble some statistics */
  return 0;
}
```

E Programme to Compute the Mean of the Dependent Variable: mean.c

```
#include <stdio.h>

/*
 * A very simple program to read in three columns X, Y, Z
 * and print out them back out again
 *
 */

int main (void)
{
  char line [256]; /* used to buffer a whole line of input */

  double in_x, in_y, in_e; /* input variables */
  double out_x, out_y, out_e; /* output variables */
  double prev_x, prev_y, prev_e; /* transitory variables */

  int counter = 0, rc; /* used for debugging bad data input */
```

```
  while (fgets (line, 256, stdin)) { /* read a whole line into buffer */
    counter++; /* increment line counter */
    rc = sscanf (line, "%lf %lf %lf", &in_x, &in_y, &in_e); /* attempt to parse variables */
    if (rc != 3) { /* if didn't find all three protest and exit */
      fprintf (stderr, "Less than three floats on line %i:\n\"%s\"\n", counter, line);
      return 1;
    }

    out_x = in_x; /* calculate  */
    out_y = in_y / in_x;
    out_e = in_e / in_x;

    printf ("%g\t%g\t%g\n", out_x, out_y, out_e); /* calculate and output result */

    prev_x = in_x; /* save the previous triplet */
    prev_y = in_y;
    prev_e = in_e;
  }
  fprintf (stderr, "Processed %i lines\n", counter); /* babble some statistics */
  return 0;
}
```