Computerphysik I: Blatt 05

Aurel Müller-Schönau und Leon Oleschko

7. Juli 2022

Aufgabe 5

a)

Appendix - Code

c)

Listing 1: ...

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


#define H 0.001
#define rstart 100


double rho(double r);
void numerov(double* y1, double* y2, double r);


int main(){

    double y1=0, y2 = 0;

    for(double r = rstart; r > 0; r -= H){
        numerov(&y1, &y2, r);
        printf("%f\n", y2);
    }

    // Ergebnis:                          -0.039789
    // Analytisch: -1/(8 pi) \approx      -0.0397887

    return 0;
}


double rho(double r){
    double wert = exp(-r)/(8 * M_PI);
    return wert;
}

void numerov(double* y1, double* y2, double r){
    double y = 2 * *y2 - *y1 - H*H/12*(rho(r+H)+10*rho(r)+rho(r-H));
    *y1 = *y2;
    *y2 = y;
}
```

f)

Listing 2: ...

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

#define H .5e-5
#define X_MAX 15.
#define N ((int) (X_MAX/H))
#define EXPORT_STEPS 1000

#define UNCERT_END .5e-15

#define UNCERT_GUESS 0.5
double guess[] = {2.5, 4.5, 6, 7.5, 8.5, 10, 12, 14, 16, 18.5};

void integ(double E);
void numerov(double *y1, double *y2, double x, double E);
void findAndExport(int n);
double V(double x);

// has to be global else segmentation fault on stack (2 MB) with (8byte/
    double * N doubles)
double phi[N];

int main(){
  // loop over guesses
  for (size_t i = 0; i < sizeof(guess)/sizeof(guess[0]); i++){
    findAndExport(i);
  }

  return 0;
}

// find the root of the error function near the n-th guess and save the
    generated wavefunction
void findAndExport(int n){
  // bisection to find the root
  double E0 = guess[n]-UNCERT_GUESS;
  double E1 = guess[n]+UNCERT_GUESS;
  double Ex, tmp;
  do {
```

```
40      Ex = (E1+E0)/2.0;
41
42      integ(Ex);
43      tmp = phi[N-1];
44      integ(E0);
45
46      if(tmp*phi[N-1]>0)
47        E0=Ex;
48      else
49        E1=Ex;
50    } while (fabs((E1-E0)/E0)>UNCERT_END);
51
52    printf("guess: %.2f E: %f\n", guess[n], E1);
53
54    // normalizing
55    double norm = 0;
56    for(int i=0; i<N; i++){
57      norm += phi[i]*phi[i]*H;
58    }
59    norm = sqrt(norm);
60    for(int i=0; i<N; i++){
61      phi[i] /= norm;
62    }
63
64    // exporting
65    FILE *file;
66    char filename[210];
67    snprintf(filename, 10, "out%02d.dat", n);
68    printf("saving in: %s\n", filename);
69    file = fopen(filename, "w+");
70    fprintf(file, "%f\n", E1);
71    for(int i=0; i<N; i+=EXPORT_STEPS){
72      fprintf(file, "%f %f\n", i*H, phi[i]);
73    }
74    fclose(file);
75  }
76
77  // integrating the wavefunction
78  void integ(double E){
79
80    double y1 = 0, y2 = H;
81    for(int i=0; i < N; i++){
82      numerov(&y1, &y2, i*H, E);
83      phi[i] = y2;
```

```
84    }
85  }
86
87  // numerov step
88  void numerov(double *y1, double *y2, double x, double E){
89
90    double y = 2* *y2*(1 - H*H*5/12*(E-V(x))) - *y1*(1 - H*H/12*(E-V(x-H)))
        ;
91    *y1 = *y2;
92    *y2 = y;
93  }
94
95
96  double V(double x){
97    return x;
98  }
```