Computer simulations in statistical physics
Homework 3: random walks

Code of program random_walk

Histogram of end-to-end distance

SAW probability

Code of program random_walk

```
#define PROGNAME "random_walk"
#define VERSION "0.3"
#define DATE  "09.11.2006"
#define AUTHOR "Nils Bluemer"

/* program random_walk: generates RWs on cartesic lattice */
/* compile with cc -lm -o random_walk random_walk.c */
/* NEW (0.2) NRRW (type 2) */
/* NEW (0.3) collision probability (~SAW rejection) */
/* TODO (0.3) generalize SAW probability for arbitrary DIM */

#include <stdlib.h>
#include <stdio.h>
#include <string.h>
#include <math.h>
#include <sys/time.h>
#define MAXSTEPS 100
#define LATDIM 201    /* 2x MAXSTEPS + 1 */
```

```
#define MIN(a,b) (a<b?a:b)
#define SQUARE(a) (a*a)

#define DIM 2

/*****************************************************************/

void random_walk (int steps, int type, int printlevel, int
coll[MAXSTEPS+1])
{
  int walker[DIM+1];
  int lattice[LATDIM][LATDIM];
  int collision;
  int i,j, n,d,l,d_old,diff,diff_old;
  double x, dist;
  int ok; /* flag for requirements */

  /*    initialization */
  for (d=1;d<=DIM;d++)
    walker[d]=0;
```

```
for (i=0;i<=2*MAXSTEPS;i++)
  for (j=0;j<=2*MAXSTEPS;j++)
    lattice[i][j]=0;
lattice[MAXSTEPS][MAXSTEPS]++;
collision=0;

diff_old=0;

for (n=0;n<=steps;){
  x=drand48();
  d=(int)(x*DIM)+1;
  x=drand48();
  if (x<0.5)
    diff=-1;
  else
    diff=1;

  if (type==1)          /* unrestricted RW */
    ok=1;
  else if (type==2){  /* nonreversal RW */
    if ((d==d_old)&&(diff+diff_old==0))
```

```
ok=0;
      else
ok=1;
    }
    if (ok==1){          /* move accepted */
      n++;
      walker[d]+=diff;
      i=walker[1]+MAXSTEPS;
      j=walker[2]+MAXSTEPS;
      lattice[i][j]++;
      if (lattice[i][j]>1)
collision++;
      if (collision>0)
coll[n]++;
      d_old=d;
      diff_old=diff;
      if (printlevel>1){
for (l=1;l<=DIM;l++)
   printf ("%3d ", walker[l]);
printf("\n");
      }
```

```
    }
  }
  if (printlevel>0){
    dist=0.0;
    for (d=1;d<=DIM;d++)
      dist+=SQUARE(walker[d]);
    dist=sqrt(dist);
    printf("# dist:  %lf\n", dist);
  }
}

void printhelp ()
{
printf("*************************************************\n");
printf("%s:  generates RWs on cartesic lattice \n",PROGNAME);
printf("Version:  %s, %s by %s\n",VERSION,DATE,AUTHOR);
printf("options:  -n# number of steps (default:  10)\n");
printf("options:  -s# sample size (default:  1)\n");
printf("          -t# type of RW (1 - unrestricted RW -
default)\n");
```

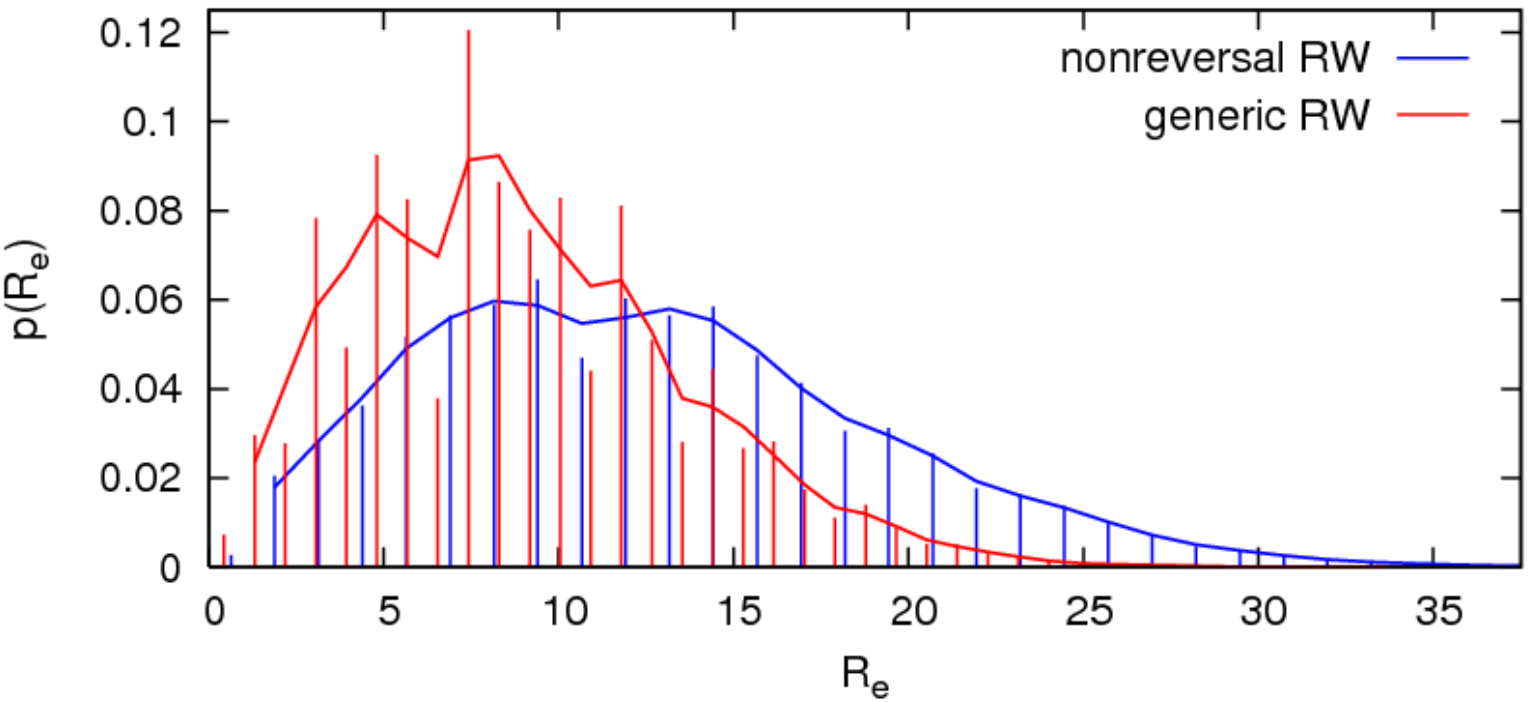
```
printf("                              (2 - nonreversal RW)\n");
printf("          -p# printlevel (0:  minimal, 1:each step)\n");
printf("          -h this help\n");
}

/*****************************************************************/

int main (int argc, char *argv[])
{
  char c;
  int n, steps, type, printlevel, samples;
  int coll[MAXSTEPS+1];
  unsigned long int i;
  struct timeval start;
  double frac;

  steps=10;
  type=1;
  printlevel=0;
  samples=1;
```

```
/* initialize RNG */
gettimeofday(&start, 0);
srand48(start.tv_usec);

while (--argc > 0 && (*++argv)[0] == '-')
  while (c= *++argv[0])
  switch (c) {
  case 'n':
     sscanf(++argv[0],"%d\n",&steps);
    if (steps>MAXSTEPS) error("steps too large");
          break;
  case 's':
     sscanf(++argv[0],"%d\n",&samples);
          break;
  case 't':
     sscanf(++argv[0],"%d\n",&type);
          break;
  case 'p':
     sscanf(++argv[0],"%d\n",&printlevel);
          break;
```

```
    case 'h':
      printhelp();
      exit(0);
    }

  for (n=1;n<=steps;n++)
    coll[n]=0;

  for(i=1;i<=samples;i++)
    random_walk(steps, type, printlevel, coll);

  printf("# %s:  generates RWs on cartesic lattice, version %s
\n",PROGNAME, VERSION);
  printf("# fraction of SAWs for RW of type %d, length %d, %d
samples\n", type, steps, samples);
  for (n=1;n<=steps;n++){
    frac=1.0-1.0*coll[n]/samples;
    printf("%d  %lf\n",n,frac);
  }
}
```

Histogram of end-to-end distance

10^5 random walks (length 100) on 2D cartesic lattice

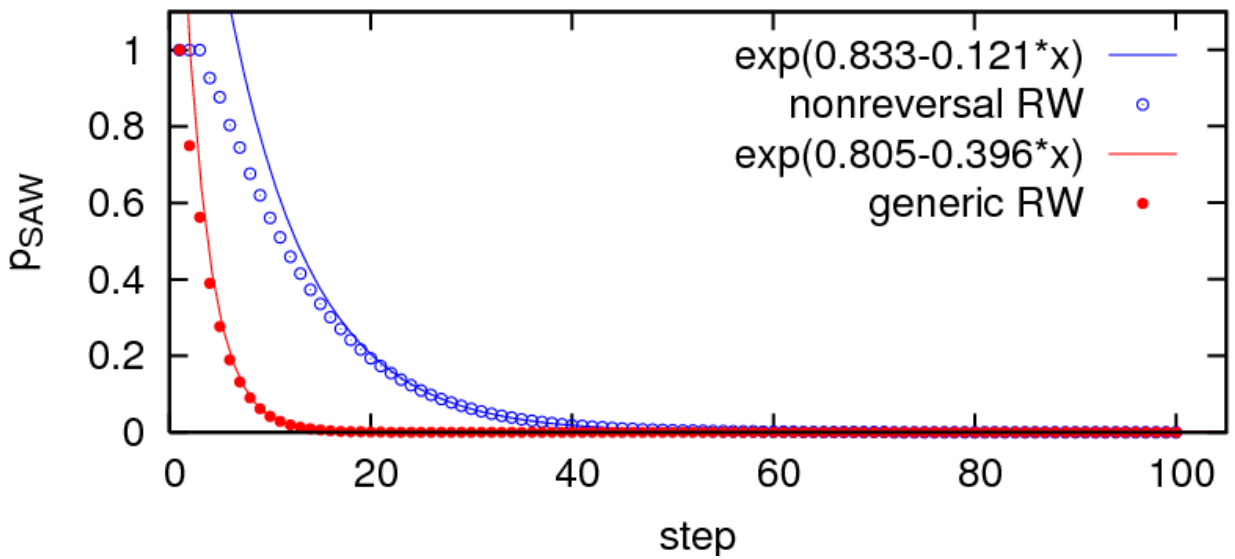


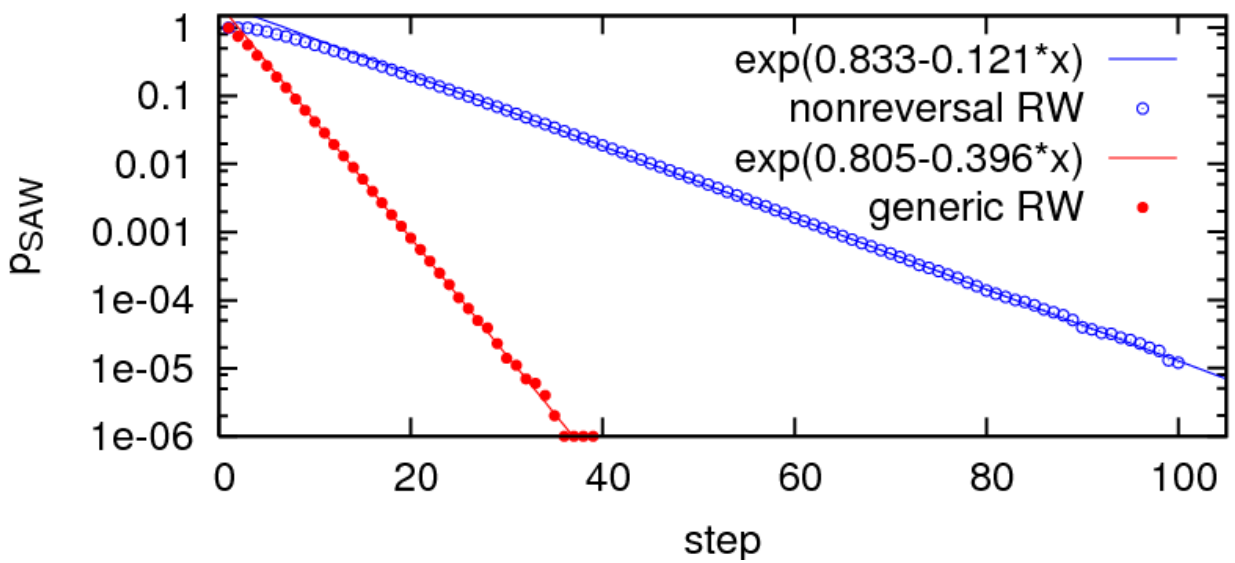
$\langle R_e \rangle \approx 8.88 \pm 0.01; \quad \sigma^2 \approx 21.5$ for generic RW

$\langle R_e \rangle \approx 12.51 \pm 0.02; \quad \sigma^2 \approx 42.0$ for nonreversal RW

roughness due to unequal spacing of possible distances (not noise)!

SAW probability





generic RW:

$p_{\mathsf{SAW}} = 1$ for step $=1$

nonreversal RW:

$p_{\mathsf{SAW}} = 1$ for step ≤ 3

$p_{\mathsf{SAW}}^{\mathsf{NNRW}} \gg p_{\mathsf{SAW}}^{\mathsf{gen}}$ for $n \gg 1$

in both cases:

exponential decay

$\rightsquigarrow$ no efficient ways to generate long SAWs