This Maple worksheet generates the data appearing in Tables 1, 2, 3, 4 and 5 in the paper "The complexity of sparse Hensel lifting and sparse polynomial factorization" by Michael Monagan and Baris Tuncer and examples for estimates in the paper.

Michael Monagan and Baris Tuncer, March 2018.

```
> restart;
```

```
> ###Create a random polynomial for examples:
  n:=7;                                   ## the number of
  variables
  deg:=15;                                 ## the total degree
  Tf:=17161;                              ## the number of terms
  X:=[seq(x[i],i=1..n)]:                    ## variables
  p:=2^31-1;                              ### prime number
  F:=randpoly(X,degree=deg,terms=Tf,coeffs = rand(1..1000)): ##
  generate a random polynomial
  Roll:=rand(1..p):                       ## random number
  generator
  sd:=binomial(n+deg,deg):                  ## possible number of
  monomials
  tF:=evalf(Tf/sd);                         ## the density ratio of
  F
```

$$n := 7$$
$$deg := 15$$
$$Tf := 17161$$
$$p := 2147483647$$
$$tF := 0.1006250586 \quad (1)$$

```
> ###to compute the number of terms of a polynomial
  numterms := proc(a)  if type(a,`+`) then nops(a) else 1 fi; end;
```

$$numterms := \mathbf{proc}(a)\ \mathbf{if}\ type(a, `+`)\ \mathbf{then}\ nops(a)\ \mathbf{else}\ 1\ \mathbf{end\ if}\ \mathbf{end\ proc} \quad (2)$$

```
> ### Eqn (3) in the paper
> TEstimate:=proc(Tf,n,d)
  local sd,s1,k;
  sd:=binomial(n+d,d);
  s1:=sum(binomial(n+k-2,k)*(1-binomial(sd-(d-k+1),Tf)/binomial(sd,
  Tf)),k=0..d);
  return evalf(s1);
  end proc:
> ##### An example
  G:=eval(F,x[n]=Roll()) mod p:   ##evaluate F mod p at a random
  point
  print("the actual numterms of G= ",numterms(G));
  print("the estimated numterms of G by (6.2) = ",TEstimate(Tf,n,
  deg));
```

```
"the actual numterms of G= ", 14359
"the estimated numterms of G by (6.2) = ", 14370.47197
```
(3)

```
> #### Theoretical estimate after m random evaluations Eqn(8)

> TEAftermEvaluationsEstimate:=proc(Tf,n,m,d)
  local sd,s1,k;
  sd:=binomial(n+d,d);
  s1:=sum(binomial(n-m-1+k,k)*(1-binomial(sd-binomial(d-k+m,m),Tf)
  /binomial(sd,Tf)),k=0..d);
  return evalf(s1);
  end proc:
> ###An example
> G:=eval(F,{x[n-2]=Roll(), x[n-1]=Roll(), x[n]=Roll()}) mod p:
  print("the actual numterms after 3 evals = ",numterms(G));
  print("the estimated numterms after 3 evals by (6.7)= ",
  TEAftermEvaluationsEstimate(Tf,n,3,deg));
                "the actual numterms after 3 evals = ", 2416
      "the estimated numterms after 3 evals by (6.7)= ", 2439.080353
```
(4)

```
> ##### Approximation to the theoretical estimate Eqn (9)

> FirstApproxEstimate:=proc(tf,n,d)
  local temp1,temp2,k;
  temp1:=binomial(n+d-1,d);
  temp2:=sum(binomial(n+k-2,k)*(1-tf)^(d-k+1),k=0..d);
  return 1.0*(temp1-temp2);
  end proc:

> ###An example
> G:=eval(F,x[n]=Roll()) mod p:
  print("the actual numterms of G = ",numterms(G));
  print("the estimated numterms of G by (6.2) = ",TEstimate(Tf,n,
```

```
deg));
print("the approx to estimate to (6.2) by (6.4) = ",
FirstApproxEstimate(tF,n,deg));
```

"the actual numterms of G = ", 14359

"the estimated numterms of G by (6.2) = ", 14370.47197

"the approx to estimate to (6.2) by (6.4) = ", 14370.36445 **(5)**

```
> ############  Example 8, Table (1) in the paper:

> n:=7;                                   ## the number of
  variables
  deg:=15;                                 ## the total degree
  Tf:=17161;                              ## the number of terms
  X:=[seq(x[i],i=1..n)]:                    ## variables
  p:=2^31-1;                              ### prime number
  F:=randpoly(X,degree=deg,terms=Tf,coeffs = rand(1..1000)):
  Roll:=rand(1..p):                       ## random number
  generator
  sd:=binomial(n+deg,deg):                  ## possible number of
  monomials
  tF:=evalf(Tf/sd);                        ## the density ratio of
  F

  G:=eval(F,x[n]=Roll()) mod p:
  nt:=numterms(F):
  tf:=evalf(nt/sd):
  Tg:=numterms(G):
  ETg:= TEstimate(Tf,n,deg):
  eTg := FirstApproxEstimate(tF,n,deg):
  tg:=evalf(Tg/binomial(n-1+degree(G),n-1)):
  Etg:=ETg/binomial(n-1+degree(G),n-1):
  etg:=eTg/binomial(n-1+degree(G),n-1):
  print("the numterms of f, Tf = ", nt);
  print("the density ratio of f, tf = ",tf);
  print("the actual numterms of g, Tg= ",Tg);
  print("the estimated numterms by (6.2) E[Tg]= ",ETg);
  print("the approx to est numterms by (6.4) eTg = ", eTg);
  print("the actual density ratio of g, tg = ", tg);
  print("the estimated density ratio of g by (6.5), E[tg] = ",etg);
  print("the approx to estimated density ratio of g by (6.5), etg =
  ", etg);
```

$$n := 7$$

$$deg := 15$$

$$Tf := 17161$$

$$p := 2147483647$$

$$tF := 0.1006250586$$

"the numterms of f, Tf = ", 17161

"the density ratio of f, tf = ", 0.1006250586

"the actual numterms of g, Tg= ", 14378

"the estimated numterms by (6.2) E[Tg]= ", 14370.47197

"the approx to est numterms by (6.4) eTg = ", 14370.36445

"the actual density ratio of g, tg = ", 0.2649638803

"the estimated density ratio of g by (6.5), E[tg] = ", 0.2648231691

"the approx to estimated density ratio of g by (6.5), etg =", 0.2648231691 **(6)**

```
> ####################Example 9, Table (2)  in the
  Paper#############
> #n:=7;
  n:=10;
  #deg:=15;
  deg:=20;
  #Tf:=17958;                    ##The number of terms to be
  guessed
  Tf:=20000;
  X:=[seq(x[i],i=1..n)]:
  Y:=X[1..n-1]:
  p:=2^31-1;
  f:=randpoly(X,degree=deg,terms=Tf,coeffs = rand(1..1000)):
  Roll:=rand(1..p):
  sd:=binomial(n+deg,deg);        ### upper bound
  tf:=evalf(Tf/sd);
```

$$n := 10$$

$$deg := 20$$

$$Tf := 20000$$

$$p := 2147483647$$

$$sd := 30045015$$

$$tf := 0.0006656678321$$ **(7)**

```
> ###  m random evaluations....
  m:=4:
  EvalPoints:={seq(x[n-i]=Roll(), i=1..m)}:
  g:=eval(f,EvalPoints) mod p:    ### What we are given if the
  numterms Tg of g
  Tg:=numterms(g):
  tg:=evalf(Tg/binomial(n-m+degree(g),degree(g))):
  gammam:=binomial(n-m+deg,deg):    ### Here we assume that there
  is no degree loss
  h:=1-(1/gammam)*sum(binomial(n-m-1+k,n-m-1)*(1-y)^binomial(deg-k+
  m,m),k=0..deg)-tg:
```

```
h2:=subs(y=1+z,h):
solution:=fsolve(h2,z,-1..0):
solution := solution +1:
E[t_f]:=solution:
E[T_f]:=E[t_f]*binomial(n+deg,deg):
print("The numterms of g, Tg = ", Tg);
print("The actual density ratio of g, tg = ", tg);
print("The guessed density ratio of f ,E[t_f] =", E[t_f]);
print("The actual density ratio of f ,t_f =", tf);
print("The guesses numterms of f, E[T_f] =", E[T_f]);
print("The actual numterms of f, T_f =", Tf);
```

"The numterms of g, Tg = ", 15016
"The actual density ratio of g, tg = ", 0.06522173479
"The guessed density ratio of f ,E[t_f] =", 0.0006711669
"The actual density ratio of f ,t_f =", 0.0006656678321
"The guesses numterms of f, E[T_f] =", 20165.21958
"The actual numterms of f, T_f =", 20000 **(8)**

```
> ############# Example 11,  Table (3) in the paper (Sparse case,
  on Zippel's assumption)
> n:=12;
  deg:=20;
  T:=10^4:
  X:=[seq(x[i],i=1..n)]:
  p:=2^31-1;
  F:=randpoly(X,degree=deg,terms=T,coeffs = rand(1..1000)):
  T:=numterms(F);
  Roll:=rand(1..p):
  sd:=binomial(n+deg,deg):
  tF:=evalf(T/sd);
```

$$n := 12$$
$$deg := 20$$
$$p := 2147483647$$
$$T := 10000$$
$$tF := 0.00004428838399$$ **(9)**

```
> f[0]:=F: tf[0]:=tF:  Tf[0]:=T:
```

```
> print("i, Tf_i, E[Tf_i], tf_i, tf_i(1+d/n-i), Tf_i/Tf_(i+1), 1+d/
```

```
(n-i+1)");
for i from 1 to n-1 do
alpha:=Roll();
f[i]:=eval(f[i-1],x[n-i+1]=alpha) mod p;
Tf[i]:=numterms(f[i]);
tf[i]:=1.0*numterms(f[i])/binomial(n-i+degree(f[i]),degree(f[i]))
;
print(i-1,Tf[i-1],TEAftermEvaluationsEstimate(Tf[0],n,i-1,deg),tf
[i-1],tf[i-1]*(1+degree(f[0])/(n-i+1)),1.0*Tf[i-1]/Tf[i],(1+1.0*
degree(f[0])/(n-i)));
od:
```

"i, Tf_i, E[Tf_i], tf_i, tf_i(1C d/n-i), Tf_i/Tf_(iC 1), 1C d/(n-iC 1)"

0, 10000, 10000., 0.00004428838399, 0.0001181023573, 1.000100010, 2.818181818

1, 9999, 9999.318749, 0.0001180905471, 0.0003328006327, 1.000500300, 3.0

2, 9994, 9995.185235, 0.0003326342157, 0.0009979026471, 1.002507774, 3.222222222

3, 9969, 9971.076950, 0.0009954063927, 0.003207420599, 1.014243565, 3.500000000

4, 9829, 9837.327859, 0.003162377075, 0.01106831976, 1.073738256, 3.857142857

5, 9154, 9200.765269, 0.01030821031, 0.03976023977, 1.282611742, 4.333333333

6, 7137, 7219.483787, 0.03099943535, 0.1343308865, 1.739458932, 5.0

7, 4103, 4144.689124, 0.07722567288, 0.3861283644, 2.398012858, 6.0

8, 1711, 1690.554432, 0.2339029392, 1.403417635, 3.422000000, 7.666666667

9, 500, 497.9369917, 0.4385964912, 3.362573099, 4.761904762, 11.0

10, 105, 104.5103466, 0.7720588235, 8.492647058, 7.500000000, 21.0 **(10)**

```
> ############# Example 12,  Table (4) in the paper (Dense case on
Zippel's assumption)
n:=7;
deg:=13;
T:=7752:
X:=[seq(x[i],i=1..n)]:
p:=2^31-1;
F:=randpoly(X,degree=deg,terms=T,coeffs = rand(1..1000)):
T:=numterms(F);
Roll:=rand(1..p):
sd:=binomial(n+deg,deg):
tF:=evalf(T/sd);
```

$$n := 7$$

$$deg := 13$$
$$p := 2147483647$$
$$T := 7752$$
$$tF := 0.1000000000 \quad \textbf{(11)}$$

```
> f[0]:=F:  tf[0]:=tF:  Tf[0]:=T:
> for i from 1 to n-1 do
  alpha:=Roll();
  f[i]:=eval(f[i-1],x[n-i+1]=alpha) mod p;
  Tf[i]:=numterms(f[i]);
  tf[i]:=1.0*numterms(f[i])/binomial(n-i+degree(f[i]),degree(f[i]))
  ;
  print(i-1,Tf[i-1],TEAftermEvaluationsEstimate(Tf[0],n,i-1,deg),tf
  [i-1],tf[i-1]*(1+degree(f[0])/(n-i+1)),1.0*Tf[i-1]/Tf[i],(1+1.0*
  degree(f[0])/(n-i)));

  od:
```

0, 7752, 7752., 0.1000000000, 0.2857142857, 1.162916292, 3.166666667

1, 6666, 6643.444284, 0.2456877488, 0.7780112045, 1.748229740, 3.600000000

2, 3813, 3800.147380, 0.4450280112, 1.602100840, 2.711948791, 4.250000000

3, 1406, 1409.834291, 0.5907563025, 2.510714286, 3.651948052, 5.333333333

4, 385, 394.0239961, 0.6875000000, 3.666666667, 4.695121951, 7.500000000

5, 82, 84.14870420, 0.9010989011, 6.758241758, 6.307692308, 14.0 **(12)**

```
> ################## Example 19 (Table 5) in the paper
> n:=7;
  deg:=13;
  r:=1.0*n/(n+deg):
  Tf:=7752:
  X:=[seq(x[i],i=1..n)];
  p:=2^31-1;
  f:=randpoly(X,degree=deg,terms=Tf,coeffs = rand(1..1000)):
  Tf:=numterms(f);
  Roll:=rand(1..p):
  sd:=binomial(n+deg,deg):
  tF:=evalf(Tf/sd);
  alpha:=Roll();
```

$$n := 7$$
$$deg := 13$$
$$X := [x_1, x_2, x_3, x_4, x_5, x_6, x_7]$$
$$p := 2147483647$$
$$Tf := 7752$$

$$tF := 0.1000000000$$

$$\alpha := 1007924763 \qquad \textbf{(13)}$$

```
> for i from 0 to deg-1 do
  cf[i]:=coeftayl(f,x[n]=alpha,i) mod p;      ## The actual i^th
  Taylor coefficient
  ntcf[i]:=numterms(cf[i]);                   ## The actual
  number of terms of the i^th Taylor Coefficient
  etf[i]:=1.0*ntcf[i]/binomial(n+deg-i,deg-i); ## The actual
  density ratio of the i^th Taylor coeff.
  Fentcf[i]:=FirstApproxEstimate(tF,n,deg-i);  ## The Estimation
  of numterms of the i^th Taylor coeff.
  BoundOnTf[i]:=tF*binomial(n+deg-i,n);       ## Bound on esti.
  of numterms of the i^th Taylor coeff.(6.15)
  if i=0 then tfder[i]:=tF;
  else
  tfder[i]:=evalf(numterms(diff(f,(x[n]$(i))))/binomial(n+deg-i,n)
  ); ## The actual dens ratio of the i^th der.
  f i :
  BoundOntf[i]:=tF*(n+deg-i)/n;             ## Bound on esti. of
  dens. ratio of the i^th Taylor coeff.(6.15)
  print(i,ntcf[i],Fentcf[i],BoundOnTf[i],tfder[i],etf[i],BoundOntf
  [i]);
  od:
```

0, 6624, 6643.345410, 7752.000000, 0.1000000000, 0.08544891641, 0.2857142857

1, 4379, 4366.828230, 5038.800000, 0.09986504723, 0.08690561245, 0.2714285714

2, 2750, 2789.364703, 3182.400000, 0.09816490699, 0.08641277024, 0.2571428571

3, 1689, 1724.183003, 1944.800000, 0.09841628959, 0.08684697655, 0.2428571429

4, 990, 1025.981115, 1144.000000, 0.09571678322, 0.08653846154, 0.2285714286

5, 563, 583.8679050, 643.5000000, 0.09588189588, 0.08749028749, 0.2142857143

6, 284, 315.0754500, 343.2000000, 0.09032634033, 0.08275058275, 0.2000000000

7, 140, 159.4171671, 171.6000000, 0.09090909091, 0.08158508158, 0.1857142857

8, 58, 74.46351900, 79.20000000, 0.08585858586, 0.07323232323, 0.1714285714

9, 28, 31.40391000, 33.00000000, 0.09393939394, 0.08484848485, 0.1571428571
10, 8, 11.55990000, 12.00000000, 0.06666666667, 0.06666666667, 0.1428571429
11, 3, 3.511000000, 3.600000000, 0.08333333333, 0.08333333333, 0.1285714286
12, 1, 0.7900000000, 0.8000000000, 0.1250000000, 0.1250000000, 0.1142857143

(14)