CONSTITUENT TEST

Magilatin generating functions and constituents

(general form, with affine data)

Notation:

L, S: magilatin, semimagic squares (all positive values).

ml: magilatin, except in g.f.'s.

l, s: normalized squares (symmetry types).

R: reduced squares (least element is 0).

r: reduced normalized squares (reduced symmetry types).

n: semimagic r.

gf: generating function in some form.

gfsum: generating function as a sum of simple terms.

c: Cubic (fixed strict upper bound; weak upper bound for reduced).

a: Affine (fixed magic sum).

p = period,

pno7 = truncated period (in affine),
p7 = period of the H term in the g.f.'s, with the denominator factor (1-x^d7).

d = degree (and dimension, in the general and no-H cases),
d7 = degree in the H term.

RtoLfactor = the rational function that multiplies Rgf to Lgf and rgf to lgf.

This is for **affine**: set up main constants.

```
> d:=4; d7:=1;
  p:=840; pno7:=120; p7:=21;
  RtoLfactor:=x^3/(1-x^3);
```

$$d := 4$$

$$d7 := 1$$

$$p := 840$$

$$pno7 := 120$$

$$p7 := 21$$

$$RtoLfactor := \frac{x^3}{1 - x^3}$$

enddegree: The number of terms of the L and l sequences to check in testing the constituents.

```
> enddegree:=2000;
```

$$enddegree := 2000$$

We start by recomputing rs from the semimagic count. From the Latte results we get the closed Ehrhart g.f. of each flat, which depends on whether we're doing cubic or affine.

This is for **affine**: set up simplex data.

```
> simplexname[1]:="OABC": ehrgf[1]:= 1/((1-x)*(1-x^2)^3) : dimen[1]:=3:
  simplexname[2]:="OEE2": ehrgf[2]:= 1/((1-x)*(1-x^4)^2) : dimen[2]:=2:
  simplexname[3]:="OAE2": ehrgf[3]:= 1/((1-x)*(1-x^2)*(1-x^4)) :
  dimen[3]:=2:
  simplexname[4]:="ADE2": ehrgf[4]:= 1/((1-x^2)*(1-x^3)*(1-x^4)) :
  dimen[4]:=2:
  simplexname[5]:="DE1E2": ehrgf[5]:= 1/((1-x^3)*(1-x^4)^2) : dimen[5]:=2:
  simplexname[6]:="OCE": ehrgf[6]:= 1/((1-x)*(1-x^2)*(1-x^4)) :
  dimen[6]:=2:
  simplexname[7]:="BDE1": ehrgf[7]:= 1/((1-x^2)*(1-x^3)*(1-x^4)) :
  dimen[7]:=2:
  simplexname[8]:="ABD": ehrgf[8]:= 1/((1-x^2)^2*(1-x^3)) : dimen[8]:=2:
  simplexname[9]:="FG1": ehrgf[9]:= 1/((1-x^5)*(1-x^8)) : dimen[9]:=1:
  simplexname[10]:="EF": ehrgf[10]:= 1/((1-x^4)*(1-x^5)) : dimen[10]:=1:
  simplexname[11]:="OG": ehrgf[11]:= 1/((1-x)*(1-x^6)) : dimen[11]:=1:
  simplexname[12]:="FG": ehrgf[12]:= 1/((1-x^5)*(1-x^6)) : dimen[12]:=1:
  simplexname[13]:="AF": ehrgf[13]:= 1/((1-x^2)*(1-x^5)) : dimen[13]:=1:
  simplexname[14]:="DG": ehrgf[14]:= 1/((1-x^3)*(1-x^6)) : dimen[14]:=1:
  simplexname[15]:="DG2": ehrgf[15]:= 1/((1-x^3)*(1-x^8)) : dimen[15]:=1:
  simplexname[16]:="DE": ehrgf[16]:= 1/((1-x^3)*(1-x^4)) : dimen[16]:=1:
  simplexname[17]:="H": ehrgf[17] := 1/(1-x^7) : dimen[17]:=0:
  # for n from 1 to 17 do print(simplexname[n], dimen[n], ehrgf[n]); od;
```

The closed E.g.f. is converted to the open E.g.f.

```
> for n from 1 to 17 do
    mu[n]:=(-1)^(dimen[1]-dimen[n]):
  od:
  mu[14]:=2*mu[14]:
  for n from 1 to 17 do
    openehrgf[n]:=simplify(-(-1)^dimen[n]*subs(x=1/x,ehrgf[n])):
  od:
```

Set up basic g.f.'s.

```
> for n from 1 to 17 do
    rsgfterm[n]:=openehrgf[n]:
  od:
  rsgf:=sum(mu[nn]*rsgfterm[nn],nn=1..17):
  sgf:=RtoLfactor*rsgf:
  rsno7gf:=sum(mu[nn]*rsgfterm[nn],nn=1..16):
  sno7gf:=RtoLfactor*rsno7gf:
  r7gf:=rsgfterm[17]:
  l7gf:=RtoLfactor*r7gf:
```

The additional faces and intersection polytopes involved in the magilatin computation. They depend on whether we're cubic or affine.

These are for **affine**.

```
> mlsimplexname[1]:="OAB": mlehrgf[1]:= 1 / ((1-x)*(1-x^2)^2) :
  mldimen[1]:=2:
  mlsimplexname[2]:="OE": mlehrgf[2]:= 1 / ((1-x)*(1-x^4)) :
  mldimen[2]:=1:
```

```
mlsimplexname[3]:="OAC": mlehrgf[3]:= 1 / ((1-x)*(1-x^2)^2) :
mldimen[3]:=2:
mlsimplexname[4]:="AD": mlehrgf[4]:= 1 / ((1-x^3)*(1-x^2)) :
mldimen[4]:=1:
mlsimplexname[5]:="DE1": mlehrgf[5]:= 1 / ((1-x^3)*(1-x^4)) :
mldimen[5]:=1:
mlsimplexname[6]:="OBC": mlehrgf[6]:= 1 / ((1-x)*(1-x^2)^2) :
mldimen[6]:=2:
mlsimplexname[7]:="OE2": mlehrgf[7]:= 1 / ((1-x)*(1-x^4)) :
mldimen[7]:=1:
mlsimplexname[8]:="BD": mlehrgf[8]:= 1 / ((1-x^2)*(1-x^3)) :
mldimen[8]:=1:
mlsimplexname[9]:="DE2": mlehrgf[9]:= 1 / ((1-x^3)*(1-x^4)) :
mldimen[9]:=1:
mlsimplexname[10]:="F": mlehrgf[10]:= 1/(1-x^5) : mldimen[10]:=0:
mlsimplexname[11]:="OB": mlehrgf[11]:= 1/((1-x)*(1-x^2)) :
mldimen[11]:=1:
# for n from 1 to 11 do print(mlsimplexname[n], mldimen[n], mlehrgf[n]);
od;
```

Now a general computation. First, open Ehrhart g.f.'s.

```
> for n from 1 to 11 do
    openmlehrgf[n]:=simplify(-(-1)^mldimen[n]*subs(x=1/x,mlehrgf[n]));
  od:
```

(-1)^3 n_OAB(1/x) equals mlehrgf[1]+mlehrgf[2], and hence n_OAB(x) is, by another method that gives a nicer appearance, summing mu(,)E^o(x):

```
> mlnnew[1] := openmlehrgf[1]-openmlehrgf[2]:
```

(-1)^3 n_OAC(1/x) equals mlehrgf[3]+mlehrgf[4]+mlehrgf[5]. Hence n_OAC(x) equals

```
> mlnnew[2] := openmlehrgf[3]-openmlehrgf[4]-openmlehrgf[5]:
```

(-1)^3 n_OBC(1/x) equals mlehrgf[6]+mlehrgf[7]+mlehrgf[8]+mlehrgf[9]+mlehrgf[10]. So n_OBC(x) equals

```
> mlnnew[3] :=
  openmlehrgf[6]-openmlehrgf[7]-openmlehrgf[8]-openmlehrgf[9]+openmlehrgf[
  10]:
```

Finally, OB gives mlehrgf[11], so that n_OB(x) is

```
> mlnnew[4] := openmlehrgf[11]:
```

To compute R, we need rs=n from semimagic, which equals rgf:

```
> Rgfsum:=72*rsgf+36*(mlnnew[1]+mlnnew[2]+mlnnew[3])+12*mlnnew[4];
  Rgf:=simplify(Rgfsum):
  Rno7gfsum:=72*rsno7gf+36*(mlnnew[1]+mlnnew[2]+mlnnew[3])+12*mlnnew[4];
  Rno7gf:=simplify(Rno7gfsum):
  R7gf:=72*r7gf:
```

$$Rgfsum := \frac{144\,x^7}{(x-1)(x^2-1)(x^4-1)} + \frac{144\,x^9}{(x^2-1)(x^3-1)(x^4-1)} + \frac{72\,x^7}{(x-1)(x^2-1)^3} + \frac{72\,x^9}{(x-1)(x^4-1)^2}$$
$$+ \frac{72\,x^{11}}{(x^3-1)(x^4-1)^2} + \frac{72\,x^7}{(x^2-1)^2(x^3-1)} + \frac{72\,x^{13}}{(x^5-1)(x^8-1)} + \frac{72\,x^9}{(x^4-1)(x^5-1)} + \frac{72\,x^7}{(x-1)(x^6-1)}$$

$$+\frac{72x^{11}}{(x^5-1)(x^6-1)}+\frac{72x^7}{(x^2-1)(x^5-1)}+\frac{144x^9}{(x^3-1)(x^6-1)}+\frac{72x^{11}}{(x^3-1)(x^8-1)}+\frac{72x^7}{x^7-1}-\frac{108x^5}{(x-1)(x^2-1)^2}$$
$$-\frac{72x^5}{(x-1)(x^4-1)}-\frac{36x^5}{x^5-1}-\frac{72x^5}{(x^3-1)(x^2-1)}+\frac{12x^3}{(x-1)(x^2-1)}$$

$$Rno7gfsum := \frac{144x^7}{(x-1)(x^2-1)(x^4-1)}+\frac{144x^9}{(x^2-1)(x^3-1)(x^4-1)}+\frac{72x^7}{(x-1)(x^2-1)^3}+\frac{72x^9}{(x-1)(x^4-1)^2}$$
$$+\frac{72x^{11}}{(x^3-1)(x^4-1)^2}+\frac{72x^7}{(x^2-1)^2(x^3-1)}+\frac{72x^{13}}{(x^5-1)(x^8-1)}+\frac{72x^9}{(x^4-1)(x^5-1)}+\frac{72x^7}{(x-1)(x^6-1)}$$
$$+\frac{72x^{11}}{(x^5-1)(x^6-1)}+\frac{72x^7}{(x^2-1)(x^5-1)}+\frac{144x^9}{(x^3-1)(x^6-1)}+\frac{72x^{11}}{(x^3-1)(x^8-1)}-\frac{108x^5}{(x-1)(x^2-1)^2}$$
$$-\frac{72x^5}{(x-1)(x^4-1)}-\frac{36x^5}{x^5-1}-\frac{72x^5}{(x^3-1)(x^2-1)}+\frac{12x^3}{(x-1)(x^2-1)}$$

Hence L, the g.f. of the number of magilatin squares, equals

```
> Lgfsum:=RtoLfactor*Rgfsum:
  Lgf:=simplify(Lgfsum):
  Lno7gfsum:=RtoLfactor*Rno7gfsum;
  Lno7gf:=simplify(Lno7gfsum):
  L7gf:=72*l7gf:
```

$$Lno7gfsum := \frac{1}{1-x^3}\left(x^3\left(\frac{144x^7}{(x-1)(x^2-1)(x^4-1)}+\frac{144x^9}{(x^2-1)(x^3-1)(x^4-1)}+\frac{72x^7}{(x-1)(x^2-1)^3}\right.\right.$$
$$+\frac{72x^9}{(x-1)(x^4-1)^2}+\frac{72x^{11}}{(x^3-1)(x^4-1)^2}+\frac{72x^7}{(x^2-1)^2(x^3-1)}+\frac{72x^{13}}{(x^5-1)(x^8-1)}+\frac{72x^9}{(x^4-1)(x^5-1)}$$
$$+\frac{72x^7}{(x-1)(x^6-1)}+\frac{72x^{11}}{(x^5-1)(x^6-1)}+\frac{72x^7}{(x^2-1)(x^5-1)}+\frac{144x^9}{(x^3-1)(x^6-1)}+\frac{72x^{11}}{(x^3-1)(x^8-1)}$$
$$\left.\left.-\frac{108x^5}{(x-1)(x^2-1)^2}-\frac{72x^5}{(x-1)(x^4-1)}-\frac{36x^5}{x^5-1}-\frac{72x^5}{(x^3-1)(x^2-1)}+\frac{12x^3}{(x-1)(x^2-1)}\right)\right)$$

Now compute the number of reduced symmetry types:

```
> rgfsum:=rsgf+mlnnew[1]+mlnnew[2]+mlnnew[3]+mlnnew[4]:
  rgf:=simplify(rgfsum):
  rno7gfsum:=rsno7gf+mlnnew[1]+mlnnew[2]+mlnnew[3]+mlnnew[4]:
  rno7gf:=simplify(rno7gfsum):
```

The g.f. of the total number of symmetry types, l_ml ("lgf"):

```
> lgfsum:=RtoLfactor*rgfsum:
  lgf:=simplify(lgfsum):
  lno7gfsum:=RtoLfactor*rno7gfsum:
  lno7gf:=simplify(lno7gfsum):
```

Generate the series expansions of the g.f.'s.

Expressing the rational function with standard denominator gives an orders-of-magnitude speedup in the series expansion.

Standard denominator (1-x^p)^{d+1}.

```
> pdenom:=(1-x^p):
  standenom:=pdenom^(d+1):
  pno7denom:=(1-x^pno7):
  stanno7denom:=pno7denom^(d+1):
  p7denom:=(1-x^p7):
  stan7denom:=p7denom^(d7+1):
```

G.f. as rational function with standard denominator.

```
> Lgfstandnum:=simplify(numer(Lgf)*simplify(standenom/denom(Lgf))):
  Lgf:=Lgfstandnum/standenom:
```

```
> Lno7gfstandnum:=simplify(numer(Lno7gf)*simplify(stanno7denom/denom(Lno7g
  f))):
  Lno7gf:=Lno7gfstandnum/stanno7denom:
```

```
> L7gfstandnum:= simplify(numer(L7gf)*simplify(stan7denom/denom(L7gf))):
  L7gf:=L7gfstandnum/stan7denom:
```

G.f. as rational function with standard denominator.

```
> Rgfstandnum:=simplify(numer(Rgf)*standenom/denom(Rgf)):
  Rgf:=Rgfstandnum/standenom:
```

G.f. as rational function with standard denominator.

```
> lgfstandnum:=simplify(numer(lgf)*simplify(standenom/denom(lgf))):
  lgf:=lgfstandnum/standenom:
```

```
> lno7gfstandnum:=simplify(numer(lno7gf)*simplify(stanno7denom/denom(lno7g
  f))):
  lno7gf:=lno7gfstandnum/stanno7denom:
```

```
> l7gfstandnum :=simplify(numer(l7gf)*simplify(stan7denom/denom(l7gf))):
  l7gf:=l7gfstandnum/stan7denom:
```

G.f. as rational function with standard denominator.

```
> rgfstandnum:=simplify(numer(rgf)*standenom/denom(rgf)):
  rgf:=rgfstandnum/standenom:
```

Expand the series to find the first few values of the number of squares.

```
> Lseries:=series(Lgf,x=0,enddegree+1):
```

Expand the series to find the first few values of the number of reduced squares.

```
> #Rseries:=series(Rgf,x=0,enddegree+1):
```

Expand the series to find the first few values of the number of symmetry types.

```
> lseries:=series(lgf,x=0,enddegree+1):
```

Expand the series to find the first few values of the number of reduced symmetry types.

```
> #rseries:=series(rgf,x=0,enddegree+1):
```

Find the constituents

First, the true 0th constitutent.

```
> Lzeroth:=expand(
  sum( coeff(Lgfstandnum,x,p*j)*binomial(d+t/p-j,d) ,j=0..d+1) );
  #print(subs(t=0,Lzeroth)):
```

$$Lzeroth := -\frac{9192}{35}t + \frac{151}{4}t^2 - 3\,t^3 + \frac{1}{8}t^4 + 948$$

Second, the truncated constituents, with no H term (denominator power 7).

Calculate the zeroth constituent of the **magilatin counting function** . Find its constant term.

```
> Lno7zeroth:=expand(
  sum(coeff(Lno7gfstandnum,x,pno7*j)*binomial(d+t/pno7-j,d),j=0..d+1) );
  #print(subs(t=0,Lno7zeroth)):
```

$$Lno7zeroth := -\frac{1296}{5}t + \frac{151}{4}t^2 - 3\,t^3 + \frac{1}{8}t^4 + 876$$

Extract the constituents of the total magilatin counting function.

```
> Lno7constituent[0]:=Lno7zeroth:
  for r from 1 to pno7  do
    Lno7constituent[r]:=expand(sum(
  coeff(Lno7gfstandnum,x,pno7*j+r)*binomial(d+(t-r)/pno7-j,d),   j=0..d)):
  #  print(r):
  #  print( Lno7constituent[r] ):
  #  print( factor(Lno7constituent[r]) ):
  od:
```

Calculate the zeroth constituent of the **truncated magilatin symmetry-type counting function** . Find its constant term.

```
> lno7zeroth:=expand(
  sum(coeff(lno7gfstandnum,x,pno7*j)*binomial(d+t/pno7-j,d),j=0..d+1) );
  #print(subs(t=0,lno7zeroth)):
```

$$lno7zeroth := -\frac{31}{15}t + \frac{25}{96}t^2 - \frac{1}{48}t^3 + \frac{1}{576}t^4 + 8$$

Extract the constituents of the truncated magilatin symmetry-type counting function.

```
> lno7constituent[0]:=lno7zeroth:
  for r from 1 to pno7  do
    lno7constituent[r]:=expand(sum(
  coeff(lno7gfstandnum,x,pno7*j+r)*binomial(d+(t-r)/pno7-j,d),   j=0..d)):
  #  print(r):
  #  print( lno7constituent[r] ):
  #  print( factor(lno7constituent[r]) ):
  od:
```

Third, the H term (the 7 part) only.

```
> L7gfstandnum: p7; d7;
```

Calculate the zeroth constituent of the **H term**. Find its constant term.

```
> L7zeroth:=expand(
  sum(coeff(L7gfstandnum,x,p7*j)*binomial(d7+t/p7-j,d7),j=0..d7+1) );
  #print(subs(t=0,L7zeroth)):
```

$$L7zeroth := -72 + \frac{24}{7} t$$

Extract the constituents of the total magilatin counting function.

```
> L7constituent[0]:=L7zeroth:
  for r from 1 to p7  do
    L7constituent[r]:=expand(sum(
  coeff(L7gfstandnum,x,p7*j+r)*binomial(d7+(t-r)/p7-j,d7),   j=0..d7 )):
  #  print(r):
  #  print( L7constituent[r] ):
  #  print( factor(L7constituent[r]) ):
  #  print( L7constituent[r]-24/7*(t-r) ):
  od:
```

Calculate the zeroth constituent of the **symmetry-type H term** . Find its constant term.

```
> l7zeroth:=expand(
  sum(coeff(numer(l7gf),x,p7*j)*binomial(d7+t/p7-j,d7),j=0..d7+1) );
  #print(subs(t=0,l7zeroth)):
```

$$l7zeroth := -1 + \frac{1}{21} t$$

Extract the constituents of the magilatin symmetry-type counting function.

```
> l7constituent[0]:=l7zeroth:
  for r from 1 to p7  do
    l7constituent[r]:=expand(sum(
  coeff(numer(l7gf),x,p7*j+r)*binomial(d7+(t-r)/p7-j,d7),   j=0..d7)):
  #  print(r):
  #  print( l7constituent[r] ):
  #  print( factor(l7constituent[r]) ):
  #  print( l7constituent[r]-1/21*(t-r) ):
  od:
```

Now the test: do the constituents give the correct numbers?

Compare the sequence to the constituent values.

```
> for n from 1 to enddegree do
   co:=coeff(Lseries,x,n):
   rno7:=modp(n,pno7);
   qno7:=(n-r)/pno7;
   r7:=modp(n,p7);
   q7:=(n-r7)/p7;
   termno7:=eval(Lno7constituent[rno7],t=n);
   term7:=eval(L7constituent[r7],t=n);
   term:=termno7-term7;
   if ( co <> term ) then printf("n= %d, r= %d, r7= %d;   no-H %d, H %d,
```

```
  seq %d, error %d \n",n ,r,r7,term,term7,co, co-termtotal) fi;
  # if ( co = term ) then print(n,rno7,r7) fi;
  od:
  print("L constituent test complete to ",enddegree);
                                "L constituent test complete to ", 2000
> for n from 1 to enddegree do
   co:=coeff(lseries,x,n):
   rno7:=modp(n,pno7);
   qno7:=(n-r)/pno7;
   r7:=modp(n,p7);
   q7:=(n-r7)/p7;
   termno7:=eval(lno7constituent[rno7],t=n);
   term7:=eval(l7constituent[r7],t=n);
   term:=termno7-term7;
   if ( co <> term ) then printf("n= %d, r= %d, r7= %d;   no-H %d, H %d,
  seq %d, error %d \n",n,r,r7,term,term7,co, co-termtotal) fi;
  # if ( co = term ) then print(n,rno7,r7) fi;
  od:
  print("l constituent test complete to ",enddegree);
                                "l constituent test complete to ", 2000
```