# r-a-delta lattice verifications

March 19, 2024

[1]:
```
using Oscar
using LinearAlgebra
using Oscar.Hecke
```

```
  ___   ____   ____    _    ____
 / _ \ / ___| / ___|  / \  |  _ \   |  Combining ANTIC, GAP, Polymake, Singular
| | | |\___ \| |     / _ \ | |_) |  |  Type "?Oscar" for more information
| |_| | ___) | |___ / ___ \|  _ <   |  Manual: https://docs.oscar-system.org
 \___/ |____/ \____/_/   \_\_| \_\  |  Version 1.0.0
```

[2]:
```
function twist(L, k)
    Lm = gram_matrix(L)
    n = minimum(size(Lm))
    Lm_twist = Lm * (k * identity_matrix(ZZ, n))
    L_twist = integer_lattice(gram = Lm_twist)
    return L_twist
end
```

[2]:
```
twist (generic function with 1 method)
```

[3]:
```
Z1 = root_lattice(:I, 1)
display(Z1)

Z2 = twist(Z1, 2)
display(Z2)

E8 = twist( root_lattice(:E, 8), -1)
display(E8)

E82 = twist(E8, 2)
display(E82)

H = hyperbolic_plane_lattice()
display(H)

H2 = twist(H, 2)
display(H2)
```

```
Integer lattice of rank 1 and degree 1
```

```
with gram matrix
[1]

Integer lattice of rank 1 and degree 1
with gram matrix
[2]

Integer lattice of rank 8 and degree 8
with gram matrix
[-2  1  0  0  0  0  0  0]
[ 1 -2  1  0  0  0  0  0]
[ 0  1 -2  1  0  0  0  1]
[ 0  0  1 -2  1  0  0  0]
[ 0  0  0  1 -2  1  0  0]
[ 0  0  0  0  1 -2  1  0]
[ 0  0  0  0  0  1 -2  0]
[ 0  0  1  0  0  0  0 -2]

Integer lattice of rank 8 and degree 8
with gram matrix
[-4  2  0  0  0  0  0  0]
[ 2 -4  2  0  0  0  0  0]
[ 0  2 -4  2  0  0  0  2]
[ 0  0  2 -4  2  0  0  0]
[ 0  0  0  2 -4  2  0  0]
[ 0  0  0  0  2 -4  2  0]
[ 0  0  0  0  0  2 -4  0]
[ 0  0  2  0  0  0  0 -4]

Integer lattice of rank 2 and degree 2
with gram matrix
[0 1]
[1 0]

Integer lattice of rank 2 and degree 2
with gram matrix
[0 2]
[2 0]
```

[4]:
```
E10 = direct_sum(H, E8)[1]
display(E10)

E10_2 = twist(E10, 2)
display(E10_2)
```

```
Integer lattice of rank 10 and degree 10
with gram matrix
[0  1  0  0  0  0  0  0  0  0]
[1  0  0  0  0  0  0  0  0  0]
[0  0 -2  1  0  0  0  0  0  0]
[0  0  1 -2  1  0  0  0  0  0]
```

```
[0   0   0   1  -2   1   0   0   0   1]
[0   0   0   0   1  -2   1   0   0   0]
[0   0   0   0   0   1  -2   1   0   0]
[0   0   0   0   0   0   1  -2   1   0]
[0   0   0   0   0   0   0   1  -2   0]
[0   0   0   0   1   0   0   0   0  -2]

Integer lattice of rank 10 and degree 10
with gram matrix
[0   2   0   0   0   0   0   0   0   0]
[2   0   0   0   0   0   0   0   0   0]
[0   0  -4   2   0   0   0   0   0   0]
[0   0   2  -4   2   0   0   0   0   0]
[0   0   0   2  -4   2   0   0   0   2]
[0   0   0   0   2  -4   2   0   0   0]
[0   0   0   0   0   2  -4   2   0   0]
[0   0   0   0   0   0   2  -4   2   0]
[0   0   0   0   0   0   0   2  -4   0]
[0   0   0   0   2   0   0   0   0  -4]
```

[11]:

```
Sdp = H2
Sen = direct_sum( twist(H, 2), twist(E8, 2) )[1]
Lnikp = direct_sum(H, H, H, twist(E8, 2) )[1]

Tdp = direct_sum(H, twist(H, 2), E8, E8)[1]
Ten = direct_sum(H, twist(H, 2), twist(E8, 2) )[1]
Lnikm = twist(E8, 2)
```

[11]:

```
Integer lattice of rank 8 and degree 8
with gram matrix
[-4   2   0   0   0   0   0   0]
[ 2  -4   2   0   0   0   0   0]
[ 0   2  -4   2   0   0   0   2]
[ 0   0   2  -4   2   0   0   0]
[ 0   0   0   2  -4   2   0   0]
[ 0   0   0   0   2  -4   2   0]
[ 0   0   0   0   0   2  -4   0]
[ 0   0   2   0   0   0   0  -4]
```

[ ]:

```
# Return whether L is primary, that is whether L is integral and its discriminant group (see discriminant_group) is a p-group for some prime number p.
# In case it is, p is also returned as second output.
#Note that for unimodular lattices, this function returns (true, 1).
# If the lattice is not primary, the second return value is -1 by default.

display( is_primary_with_prime(H) ) #unimodular
display( is_primary_with_prime(E8) ) #unimodular
```

```
display( is_primary_with_prime(E10) ) #unimodular
display( is_primary_with_prime(E10_2) ) #2-primary
```

[ ]:
```
display( is_elementary_with_prime(H) ) #unimodular
display( is_elementary_with_prime(E8) ) #unimodular
display( is_elementary_with_prime(E10) ) #unimodular
display( is_elementary_with_prime(E10_2) ) #2-elementary
```

[ ]:
```
# Return the number of (positive, zero, negative) inertia of L.

display(signature_tuple(H))
display(signature_tuple(E8))
display(signature_tuple(E10))
display(signature_tuple(E10_2))
```

[ ]:
```
discriminant_group(E10)
```

[ ]:
```
discriminant_group(E10_2)
```

[ ]:
```

```

[ ]:
```
display( signature_tuple(L) )
display( is_primary_with_prime(L) )
display( is_elementary_with_prime(L) )
display( discriminant_group(L) )
```

# 1 Coparity

AE22: Compactifications of moduli spaces of K3 surfaces with a nonsymplectic involution

**Definition 2.3.** We define an additional invariant, coparity $\delta_H$ as follows: $\delta = 0$ if for all $x \in A_H$ one has $q_H(x) \equiv 0(\mathrm{mod}\mathbb{Z})$ and $\delta = 1$ otherwise. We will call lattices with $\delta_H = 0$ co-even and lattices with $\delta_H = 1$ co-odd.

[77]:
```
function rad_invts(lat::ZZLat)
    display("------------------------------------")
    is_elem = is_elementary_with_prime(lat)
    if is_elem[1] == false || is_elem[2] != 2
        display("Not a 2-elementary lattice:")
        display(is_elem)
        return 0
    end
    #display("This is a p-elementary lattice.")

    D_L = discriminant_group(lat)
    display(D_L)
    Q = D_L.gram_matrix_quadratic
    G = D_L.ab_grp
```

```
    #display(G)

    #display("Computing r..")
    r = rank(lat)

    #display("Computing a..")
    a = length( filter(x -> x == 2, elementary_divisors(G)) )

    #display("Computing delta...")

    n = minimum(size(Q))
    diags = [ Q[i, i] for i in 1:n ]
    #show("Diagonal of Q:")
    #show(diags)
    are_diags_ints = map(is_integer, diags)
    all_integer_diags = reduce(&, are_diags_ints)
    # = 1 if all integers, = 0 if any non-integer

    delta = 1 - all_integer_diags
    # delta = 0 <=> image in Z, dellta=1 <=> non-integral image

    display("------------------------------------")

    return (r, a, delta)
end
```

[77]: rad_invts (generic function with 1 method)

[78]:
```
display( rad_invts(Sdp) )
display( rad_invts( Sen ) )
display( rad_invts( Lnikp ) )

display( rad_invts( Tdp ) )
display( rad_invts( Ten ) )
display( rad_invts( Lnikm ) )
```

```
"------------------------------------"

Finite quadratic module
  over integer ring
Abelian group: (Z/2)^2
Bilinear value module: Q/Z
Quadratic value module: Q/2Z
Gram matrix quadratic form:
[   0   1//2]
[1//2      0]

"------------------------------------"
```

```
(2, 2, 0)

"------------------------------------"

Finite quadratic module
  over integer ring
Abelian group: (Z/2)^10
Bilinear value module: Q/Z
Quadratic value module: Q/2Z
Gram matrix quadratic form:
[   0 1//2    0    0    0    0    0    0    0    0]
[1//2    0    0    0    0    0    0    0    0    0]
[   0    0    0 1//2    0    0    0    0    0 1//2]
[   0    0 1//2    1    0    0    0    0    0    0]
[   0    0    0    0    1    0    0    0    0 1//2]
[   0    0    0    0    0    0 1//2    0 1//2    0]
[   0    0    0    0    0 1//2    0    0    0 1//2]
[   0    0    0    0    0    0    0    1 1//2    0]
[   0    0    0    0    0 1//2    0 1//2    1 1//2]
[   0    0 1//2    0 1//2    0 1//2    0 1//2    0]

"------------------------------------"

(10, 10, 0)

"------------------------------------"

Finite quadratic module
  over integer ring
Abelian group: (Z/2)^8
Bilinear value module: Q/Z
Quadratic value module: Q/2Z
Gram matrix quadratic form:
[   0 1//2    0    0    0    0    0 1//2]
[1//2    1    0    0    0    0    0    0]
[   0    0    1    0    0    0    0 1//2]
[   0    0    0    0 1//2    0 1//2    0]
[   0    0    0 1//2    0    0    0 1//2]
[   0    0    0    0    0    1 1//2    0]
[   0    0    0 1//2    0 1//2    1 1//2]
[1//2    0 1//2    0 1//2    0 1//2    0]

"------------------------------------"

(14, 8, 0)

"------------------------------------"

Finite quadratic module
  over integer ring
Abelian group: (Z/2)^2
Bilinear value module: Q/Z
Quadratic value module: Q/2Z
```

```
Gram matrix quadratic form:
[   0   1//2]
[1//2      0]
```

```
"-----------------------------------"
```

```
(20, 2, 0)
```

```
"-----------------------------------"
```

```
Finite quadratic module
  over integer ring
Abelian group: (Z/2)^10
Bilinear value module: Q/Z
Quadratic value module: Q/2Z
Gram matrix quadratic form:
[   0   1//2      0      0      0      0      0      0      0      0]
[1//2      0      0      0      0      0      0      0      0      0]
[   0      0      0   1//2      0      0      0      0      0   1//2]
[   0      0   1//2      1      0      0      0      0      0      0]
[   0      0      0      0      1      0      0      0      0   1//2]
[   0      0      0      0      0      0   1//2      0   1//2      0]
[   0      0      0      0      0   1//2      0      0      0   1//2]
[   0      0      0      0      0      0      0      1   1//2      0]
[   0      0      0      0      0   1//2      0   1//2      1   1//2]
[   0      0   1//2      0   1//2      0   1//2      0   1//2      0]
```

```
"-----------------------------------"
```

```
(12, 10, 0)
```

```
"-----------------------------------"
```

```
Finite quadratic module
  over integer ring
Abelian group: (Z/2)^8
Bilinear value module: Q/Z
Quadratic value module: Q/2Z
Gram matrix quadratic form:
[   0   1//2      0      0      0      0      0   1//2]
[1//2      1      0      0      0      0      0      0]
[   0      0      1      0      0      0      0   1//2]
[   0      0      0      0   1//2      0   1//2      0]
[   0      0      0   1//2      0      0      0   1//2]
[   0      0      0      0      0      1   1//2      0]
[   0      0      0   1//2      0   1//2      1   1//2]
[1//2      0   1//2      0   1//2      0   1//2      0]
```

```
"-----------------------------------"
```

```
(8, 8, 0)
```

[86]:

```
#display( rad_invts( E10 ) )
#display( rad_invts( twist(E10, 2) ) )
```

[85]:

```
#L = direct_sum(Z2, H2, E82)[1]
#display( rad_invts(L) )
```

[ ]:

```
```