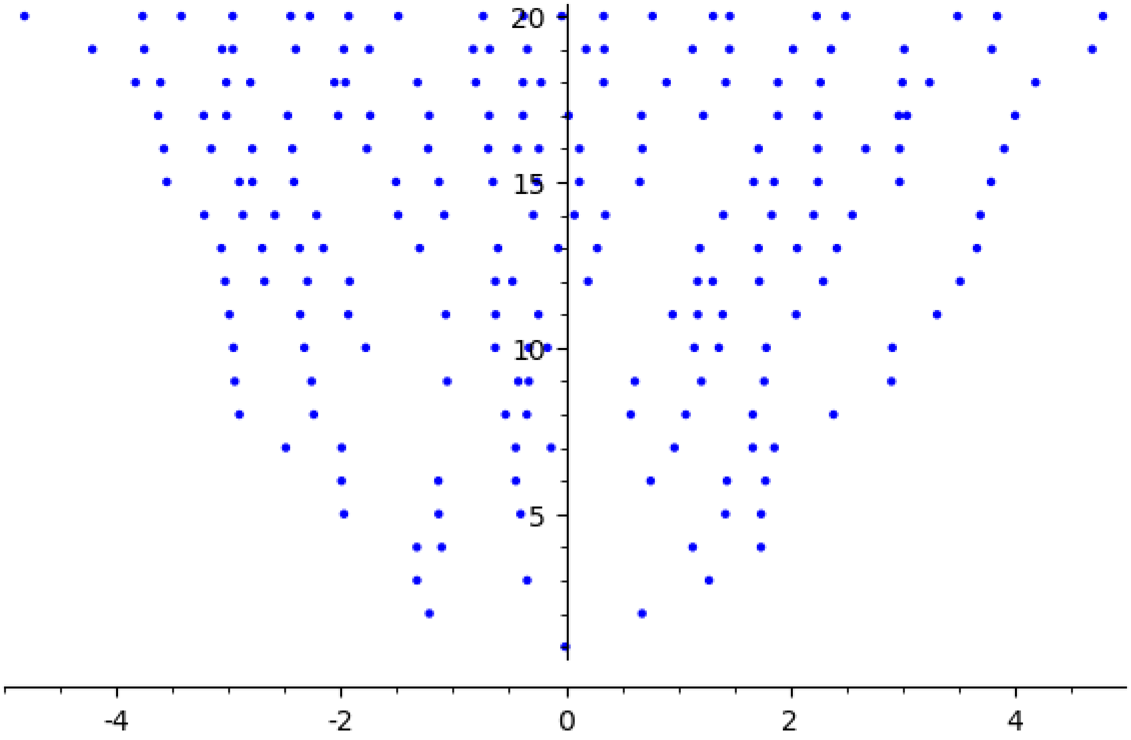
In [345]:

```
# Problem 1

reset()
def RSM(n):
    A = random_matrix(RDF,n)
    for i in range(n):
        for j in range(n):
            if i < j:
                A[i,j] = A[j,i]
    return A

def evalArray(d):
    result = []
    M = RSM(d)
    for i in range(1,d+1):
        e = M[:i,:i].eigenvalues()
        for j in e:
            result.append([j,i])
    return result

show(RSM(5))
show(list_plot(evalArray(20)))
```

$$\left(\begin{array}{rrrrr}
0.9676918479067342 & 0.44544128307314423 & -0.6302275749992177 & 0.3830493535827537 & 0.5037 \\
0.44544128307314423 & -0.012474995964299351 & -0.6863382161192515 & -0.49015932772249715 & 0.299 \\
-0.6302275749992177 & -0.6863382161192515 & 0.2848492849330444 & 0.07194002548839973 & -0.5781 \\
0.3830493535827537 & -0.49015932772249715 & 0.07194002548839973 & 0.6590391668975091 & -0.10342 \\
0.5037070363745717 & 0.299734081109597 & -0.5781994441666509 & -0.10342356969359501 & 0.17854
\end{array}\right.$$



In [346]:

```
# Problem 2

reset()
def myMaj(perm):
    result = 0
    for i in range(len(perm)-1):
        if perm[i] > perm[i+1]:
            result += i+1
    return result

def myInv(perm):
    counter = 0
    for i in range(len(perm)):
        for j in range(i):
            if(perm[j] > perm[i]):
                counter += 1
    return counter

def majorArray(d):
    p = Permutations(range(d)).list()
    result = [0] * (myMaj(p[-1])+1)
    for i in range(len(p)):
        result[myMaj(p[i])] += 1
    return result

def invArray(d):
    p = Permutations(range(d)).list()
    result = [0] * (myMaj(p[-1])+1)
    for i in range(len(p)):
        result[myInv(p[i])] += 1
    return result



# print(myMaj([1,4,2,5,3]))
print(myMaj([1,3,2,5,4]))
# print(myInv([1,4,2,5,3]))
print(myInv([1,3,2,5,4]))
# print(majorArray(4))
# print(invArray(5))
print(majorArray(7))
print(invArray(7))
```

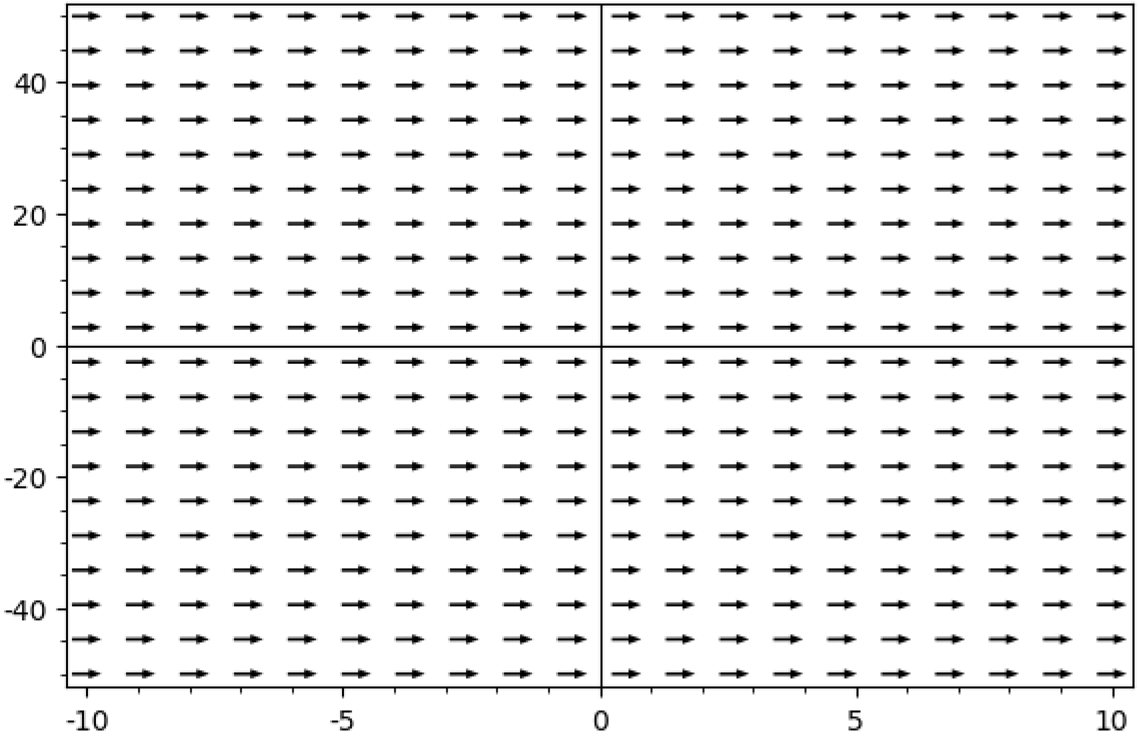
```
6
2
[1, 6, 20, 49, 98, 169, 259, 359, 455, 531, 573, 573, 531, 455, 359, 259, 169, 98, 49, 20, 6, 1]
[1, 6, 20, 49, 98, 169, 259, 359, 455, 531, 573, 573, 531, 455, 359, 259, 169, 98, 49, 20, 6, 1]
```

In [347]:

```
# Problem 3

# a
var('p,t')
y = function('y')(t)
myDiffEq = diff(y,t) == p*y*(log(35000)-log(y))
# show(myDiffEq)
a = desolve(myDiffEq,y,ivar=t,ics=[5,4])
# show(a)
b = solve(a,p)
# show(b)
d = b[0].subs({y(0):1,t:0})
show(d)

# b
e = myDiffEq.subs(p=d.rhs()).rhs()
sf = plot_slope_field(d.rhs().n(),(t,-10,10),(y,-50,50),headlength=4,headaxislength=4)
plotsol = plot(e,(y,-10,10),ymin=-50,ymax=50,color='red')
show(sf+plotsol)

# c
show(solve(a.subs({y(t):6250,p:d.rhs()}),t))

# d
# show(a.subs(p=d.rhs()).rhs())
# e.find_root(0,35000)

# e
show(solve(a.subs({y(t):4000,p:d.rhs()}),t)[0].rhs() - solve(a.subs({y(t):2000,p:d.rhs()}),t)[0].rhs())
```

$$p = \frac{1}{5}\log(-\log(35000)) - \frac{1}{5}\log(-\log(35000) + 2\,\log(2))$$

```
verbose 0 (3797: plot.py, generate_plot_points) WARNING: When plotting, failed to evaluate function at 200 po
ints.
verbose 0 (3797: plot.py, generate_plot_points) Last error message: 'unable to simplify to float approximatio
n'
```



$$\left[t = \frac{5\,(\log(-\log(35000)) - \log(-\log(35000) + \log(6250)))}{\log(-\log(35000)) - \log(-\log(35000) + 2\,\log(2))}\right]$$

$$\frac{5\,(\log(-\log(35000)) - \log(-\log(35000) + \log(4000)))}{\log(-\log(35000)) - \log(-\log(35000) + 2\,\log(2))} - \frac{5\,(\log(-\log(35000)) - \log(-\log(35000) + \log(2000)))}{\log(-\log(35000)) - \log(-\log(35000) + 2\,\log(2))}$$