作业2 SA19225404 吴语港

写代码生成一个包含200条数据的数据集（数据集的具体要求见步骤1））， 再按照步骤2）~步骤4）的要求生成指定模型，训练模型、 并分别画出训练过程中模型在训练集和验证集上的学习曲线， 利用学习曲线判断模型是欠拟合状态、过拟合状态，还是表现正好。

In [35]:
```
#准备工作
import d2lzh as d2l
#numpy
import numpy as np
#数据预处理
from sklearn import preprocessing
#需要先pip好mxnet，再导入gluon，nd
from mxnet import autograd, gluon, nd
#导入数据，起小名
from mxnet.gluon import data as gdata, loss as gloss, nn
```

1）生成一个人工数据集。在训练数据集和测试数据集中，给定样本特征x，我们使用如下的三阶多项式函数来生成该样本的标签： $y = 1.2x - 3.4x2 + 5.6x3 + 5 + \epsilon$ 其中噪声项ϵ服从均值为0、标准差为0.1的正态分布。训练数据集和测试数据集的样本数都设为100。

In [49]:
```
#训练集大小，测试集大小，多项式的系数（权重数组），常数项偏置
n_train, n_test, true_w, true_b = 100, 100, [1.2, -3.4, 5.6], 5
#【200x1矩阵】一维随机数正态分布，参数初始化，做输入x也就是features，一阶拟合
features = nd.random.normal(shape=(n_train + n_test, 1))
#【200x3矩阵】一次项，二次项，三次项，不带系数（权重），可以看成中间变量，三阶拟合
poly_features = nd.concat(features, nd.power(features, 2),nd.power(features, 3
))
#【200x10矩阵】十阶拟合使用，容易nan，需要标准化或者调低学习率
ten_features = nd.concat(features, nd.power(features, 2),nd.power(features, 3
),
                         nd.power(features, 4),nd.power(features, 5),
                         nd.power(features, 6),nd.power(features, 7),
                         nd.power(features, 8),nd.power(features, 9),nd.power(
features,10))
#标准化
# a = nd.array(ten_features, dtype=np.float64)
# a = np.concatenate(a, axis=0)
# preprocessing.scale(a)
#【1x200矩阵】三阶多项式函数生成数据，带上系数计算，并加上常数偏置，进行求和
labels = (true_w[0] * poly_features[:, 0] + true_w[1] * poly_features[:, 1]
          + true_w[2] * poly_features[:, 2] + true_b)
#【1x200】加上方差为0.1的正态分布噪声
labels += nd.random.normal(scale=0.1, shape=labels.shape)
```

In [37]:
```
#查看样本全体，输入与输出
features[:], poly_features[:],ten_features[:], labels[:]
```

Out[37]:
```
(
 [[ 0.12576538]
  [-0.22553162]
  [-0.08724168]
  [-1.9982861 ]
  [ 0.2652651 ]
```

```
[ 0.7415213 ]
[-1.1023985 ]
[ 0.4312621 ]
[ 0.04796332]
[ 1.0383999 ]
[-0.6669437 ]
[ 0.536152  ]
[ 0.85688543]
[-1.1319188 ]
[-0.70708483]
[ 0.83172685]
[-0.3453634 ]
[ 1.4978392 ]
[ 0.642728  ]
[-0.01772865]
[-2.7624755 ]
[ 0.70027727]
[-0.40241793]
[-0.09507764]
[-1.0602223 ]
[-0.0284882 ]
[ 1.4242371 ]
[ 2.8982308 ]
[-1.2253977 ]
[-0.84328055]
[ 0.88106   ]
[ 0.1136774 ]
[-1.609424  ]
[-0.41303247]
[ 2.2879524 ]
[-0.88596725]
[-0.83625084]
[ 0.8604862 ]
[ 0.9609362 ]
[-0.685047  ]
[-0.25356722]
[ 0.8301784 ]
[ 0.8131436 ]
[-0.24233957]
[ 0.800346  ]
[ 0.47322297]
[-0.05364884]
[-1.5491834 ]
[ 0.09825653]
[ 0.29727766]
[-0.8730638 ]
[ 2.0189908 ]
[-0.728708  ]
[-0.6191371 ]
[-0.06395611]
[-2.3897045 ]
[-0.3885556 ]
[ 2.0347214 ]
[-0.25581995]
[ 0.3952196 ]
[ 1.1336962 ]
[-0.8509516 ]
[-0.0282831 ]
[-0.6499274 ]
[ 0.20007886]
[ 2.1012967 ]
                ]
```

Loading [MathJax]/jax/output/HTML-CSS/jax.js

```
[ 1.1521629 ]
[ 0.08766306]
[ 0.51071817]
[ 0.14053634]
[ 0.9358318 ]
[ 0.11389208]
[-0.14931256]
[-1.4150996 ]
[-0.87655705]
[-0.16630606]
[-0.8371979 ]
[ 0.26143324]
[ 1.2776461 ]
[ 0.28967476]
[-1.2404923 ]
[-0.07493219]
[-2.1264658 ]
[-0.43312496]
[ 0.3131587 ]
[-0.4197384 ]
[ 0.8411091 ]
[ 0.8828689 ]
[-0.71184736]
[-0.96728325]
[-1.3908057 ]
[-0.00522797]
[-1.4671829 ]
[-1.1560943 ]
[-0.28207478]
[ 1.5991997 ]
[-1.0981686 ]
[ 0.08028393]
[-0.59650564]
[-1.7118372 ]
[ 0.8820606 ]
[ 1.3208119 ]
[-0.78411746]
[-1.0087717 ]
[ 1.5995713 ]
[ 2.1544929 ]
[-0.05950052]
[ 0.43422657]
[-0.600691  ]
[ 0.05431196]
[ 0.22879006]
[ 1.0659082 ]
[-1.7202955 ]
[ 0.28747115]
[ 1.9047385 ]
[ 1.5237433 ]
[ 0.9028696 ]
[ 0.05954906]
[-0.1553183 ]
[-1.2079413 ]
[ 0.5892751 ]
[ 0.65562326]
[ 0.20619893]
[-0.17643832]
[-0.05087833]
[-1.0953387 ]
[-0.5711226 ]
```

```
[-1.0478936 ]
[-0.9163434 ]
[-1.3336002 ]
[ 0.636083  ]
[-0.33219483]
[ 1.2063538 ]
[-0.50075674]
[ 0.9667549 ]
[-0.46244988]
[ 0.27516472]
[ 0.03172834]
[-2.2662334 ]
[-0.1055946 ]
[ 0.700264  ]
[ 0.30801338]
[-0.5782221 ]
[-0.20607154]
[ 0.693279  ]
[ 0.6269737 ]
[-0.35832733]
[ 0.48003122]
[ 1.5261042 ]
[-0.01285097]
[ 0.88687867]
[-0.31120738]
[ 1.1873251 ]
[ 0.3758569 ]
[ 0.31230175]
[-0.43884483]
[ 0.5022598 ]
[-0.6357437 ]
[ 1.1511858 ]
[-0.16127418]
[ 0.34598973]
[-0.5449143 ]
[-0.55088896]
[-0.53049916]
[-0.41114143]
[ 0.2638282 ]
[-0.6453085 ]
[ 0.68439865]
[-0.04890326]
[ 0.02328648]
[-0.16014808]
[-0.7801847 ]
[ 0.12058286]
[ 0.7193251 ]
[ 0.16335344]
[ 0.46283942]
[-1.5241615 ]
[ 0.0095202 ]
[ 2.3447704 ]
[ 1.4452322 ]
[ 1.6769009 ]
[-0.6372913 ]
[ 0.4833494 ]
[-0.01161444]
[ 0.9083601 ]
[ 1.2296662 ]
[ 0.24398364]
[ 1.5268717 ]
```

```
 [ 0.94724864]
 [ 0.47048146]
 [ 1.4172789 ]
 [ 1.0515909 ]
 [-0.19438495]
 [ 0.32892177]
 [ 1.8646097 ]
 [-0.818179  ]]
<NDArray 200x1 @cpu(0)>,
[[ 1.25765383e-01  1.58169325e-02  1.98922236e-03]
 [-2.25531623e-01  5.08645140e-02 -1.14715565e-02]
 [-8.72416794e-02  7.61111081e-03 -6.64006046e-04]
 [-1.99828613e+00  3.99314737e+00 -7.97945118e+00]
 [ 2.65265107e-01  7.03655779e-02  1.86655317e-02]
 [-5.38095236e-01  2.89546490e-01 -1.55803576e-01]
 [ 7.41521299e-01  5.49853861e-01  4.07728344e-01]
 [-1.10239851e+00  1.21528244e+00 -1.33972561e+00]
 [ 4.31262106e-01  1.85987011e-01  8.02091435e-02]
 [ 4.79633212e-02  2.30048015e-03  1.10338668e-04]
 [ 1.03839993e+00  1.07827437e+00  1.11968005e+00]
 [-6.66943729e-01  4.44813937e-01 -2.96665877e-01]
 [ 5.36152005e-01  2.87458986e-01  1.54121712e-01]
 [ 8.56885433e-01  7.34252632e-01  6.29170418e-01]
 [-1.13191879e+00  1.28124011e+00 -1.45025980e+00]
 [-7.07084835e-01  4.99968976e-01 -3.53520483e-01]
 [ 8.31726849e-01  6.91769540e-01  5.75363338e-01]
 [-3.45363408e-01  1.19275883e-01 -4.11935262e-02]
 [ 1.49783921e+00  2.24352241e+00  3.36043572e+00]
 [ 6.42727971e-01  4.13099259e-01  2.65510440e-01]
 [-1.77286454e-02  3.14304867e-04 -5.57219937e-06]
 [-2.76247549e+00  7.63127089e+00 -2.10811977e+01]
 [ 7.00277269e-01  4.90388244e-01  3.43407750e-01]
 [-4.02417928e-01  1.61940187e-01 -6.51676357e-02]
 [-9.50776413e-02  9.03975777e-03 -8.59478838e-04]
 [-1.06022227e+00  1.12407124e+00 -1.19176543e+00]
 [-2.84881983e-02  8.11577425e-04 -2.31203794e-05]
 [ 1.42423713e+00  2.02845144e+00  2.88899589e+00]
 [ 2.89823079e+00  8.39974213e+00  2.43443909e+01]
 [-1.22539771e+00  1.50159955e+00 -1.84005666e+00]
 [-8.43280554e-01  7.11122096e-01 -5.99675417e-01]
 [ 8.81060004e-01  7.76266754e-01  6.83937550e-01]
 [ 1.13677405e-01  1.29225524e-02  1.46900222e-03]
 [-1.60942400e+00  2.59024549e+00 -4.16880322e+00]
 [-4.13032472e-01  1.70595825e-01 -7.04616159e-02]
 [ 2.28795242e+00  5.23472643e+00  1.19768047e+01]
 [-8.85967255e-01  7.84937978e-01 -6.95429325e-01]
 [-8.36250842e-01  6.99315488e-01 -5.84803164e-01]
 [ 8.60486209e-01  7.40436494e-01  6.37135386e-01]
 [ 9.60936189e-01  9.23398376e-01  8.87326896e-01]
 [-6.85046971e-01  4.69289362e-01 -3.21485251e-01]
 [-2.53567219e-01  6.42963350e-02 -1.63034424e-02]
 [ 8.30178380e-01  6.89196169e-01  5.72155714e-01]
 [ 8.13143611e-01  6.61202550e-01  5.37652612e-01]
 [-2.42339566e-01  5.87284639e-02 -1.42322313e-02]
 [ 8.00346017e-01  6.40553772e-01  5.12664616e-01]
 [ 4.73222971e-01  2.23939985e-01  1.05973542e-01]
 [-5.36488406e-02  2.87819817e-03 -1.54411988e-04]
 [-1.54918337e+00  2.39996910e+00 -3.71799231e+00]
 [ 9.82565284e-02  9.65434499e-03  9.48602450e-04]
 [ 2.97277659e-01  8.83740038e-02  2.62716170e-02]
 [-8.73063803e-01  7.62240410e-01 -6.65484488e-01]
              e+00  4.07632351e+00  8.23005962e+00]
```

Loading [MathJax]/jax/output/HTML-CSS/jax.js

```
 [-7.28708029e-01  5.31015396e-01 -3.86955172e-01]
 [-6.19137108e-01  3.83330762e-01 -2.37334296e-01]
 [-6.39561117e-02  4.09038411e-03 -2.61605077e-04]
 [-2.38970447e+00  5.71068764e+00 -1.36468554e+01]
 [-3.88555586e-01  1.50975451e-01 -5.86623512e-02]
 [ 2.03472137e+00  4.14009094e+00  8.42393208e+00]
 [-2.55819947e-01  6.54438436e-02 -1.67418402e-02]
 [ 3.95219594e-01  1.56198531e-01  6.17327206e-02]
 [ 1.13369620e+00  1.28526711e+00  1.45710242e+00]
 [-8.50951612e-01  7.24118650e-01 -6.16189957e-01]
 [-2.82830987e-02  7.99933681e-04 -2.26246029e-05]
 [-6.49927378e-01  4.22405601e-01 -2.74532974e-01]
 [ 2.00078860e-01  4.00315486e-02  8.00946727e-03]
 [ 2.10129666e+00  4.41544771e+00  9.27816582e+00]
 [-1.04035389e+00  1.08233619e+00 -1.12601268e+00]
 [ 1.15216291e+00  1.32747936e+00  1.52947247e+00]
 [ 8.76630619e-02  7.68481242e-03  6.73674163e-04]
 [ 5.10718167e-01  2.60833055e-01  1.33212179e-01]
 [ 1.40536338e-01  1.97504628e-02  2.77565769e-03]
 [ 9.35831785e-01  8.75781119e-01  8.19583833e-01]
 [ 1.13892078e-01  1.29714059e-02  1.47734035e-03]
 [-1.49312556e-01  2.22942401e-02 -3.32880975e-03]
 [-1.41509962e+00  2.00250697e+00 -2.83374691e+00]
 [-8.76557052e-01  7.68352270e-01 -6.73504591e-01]
 [-1.66306064e-01  2.76577063e-02 -4.59964434e-03]
 [-8.37197900e-01  7.00900316e-01 -5.86792290e-01]
 [ 2.61433244e-01  6.83473423e-02  1.78682674e-02]
 [ 1.27764606e+00  1.63237941e+00  2.08560324e+00]
 [ 2.89674759e-01  8.39114636e-02  2.43070330e-02]
 [-1.24049234e+00  1.53882122e+00 -1.90889597e+00]
 [-7.49321878e-02  5.61483297e-03 -4.20731696e-04]
 [-2.12646580e+00  4.52185678e+00 -9.61557388e+00]
 [-4.33124959e-01  1.87597230e-01 -8.12530443e-02]
 [ 3.13158691e-01  9.80683640e-02  3.07109617e-02]
 [-4.19738412e-01  1.76180333e-01 -7.39496574e-02]
 [ 8.41109097e-01  7.07464516e-01  5.95054865e-01]
 [ 8.82868886e-01  7.79457450e-01  6.88158751e-01]
 [-7.11847365e-01  5.06726682e-01 -3.60712051e-01]
 [-9.67283249e-01  9.35636878e-01 -9.05025899e-01]
 [-1.39080572e+00  1.93434060e+00 -2.69029188e+00]
 [-5.22797368e-03  2.73317091e-05 -1.42889448e-07]
 [-1.46718287e+00  2.15262556e+00 -3.15829539e+00]
 [-1.15609431e+00  1.33655405e+00 -1.54518259e+00]
 [-2.82074779e-01  7.95661807e-02 -2.24436130e-02]
 [ 1.59919965e+00  2.55743957e+00  4.08985662e+00]
 [-1.09816861e+00  1.20597434e+00 -1.32436311e+00]
 [ 8.02839324e-02  6.44550985e-03  5.17470879e-04]
 [-5.96505642e-01  3.55818987e-01 -2.12248027e-01]
 [-1.71183717e+00  2.93038654e+00 -5.01634455e+00]
 [ 8.82060587e-01  7.78030872e-01  6.86270356e-01]
 [ 1.32081187e+00  1.74454403e+00  2.30421448e+00]
 [-7.84117460e-01  6.14840209e-01 -4.82106924e-01]
 [-1.00877166e+00  1.01762021e+00 -1.02654648e+00]
 [ 1.59957135e+00  2.55862856e+00  4.09270906e+00]
 [ 2.15449286e+00  4.64183950e+00  1.00008097e+01]
 [-5.95005192e-02  3.54031171e-03 -2.10650396e-04]
 [ 4.34226573e-01  1.88552722e-01  8.18746015e-02]
 [-6.00691020e-01  3.60829711e-01 -2.16747165e-01]
 [ 5.43119572e-02  2.94978870e-03  1.60208801e-04]
 [ 2.28790060e-01  5.23448922e-02  1.19759906e-02]
 [ 1.06590819e+00  1.13616025e+00  1.21104252e+00]
                e+00  2.95941687e+00 -5.09107161e+00]
```

```
 [ 2.87471145e-01  8.26396570e-02  2.37565171e-02]
 [ 1.90473855e+00  3.62802887e+00  6.91044664e+00]
 [ 1.52374327e+00  2.32179356e+00  3.53781724e+00]
 [ 9.02869582e-01  8.15173507e-01  7.35995352e-01]
 [ 5.95490597e-02  3.54609056e-03  2.11166349e-04]
 [-1.55318305e-01  2.41237767e-02 -3.74686392e-03]
 [-1.20794129e+00  1.45912218e+00 -1.76253390e+00]
 [ 5.89275122e-01  3.47245157e-01  2.04622939e-01]
 [ 6.55623257e-01  4.29841846e-01  2.81814307e-01]
 [ 2.06198931e-01  4.25179973e-02  8.76716617e-03]
 [-1.76438317e-01  3.11304796e-02 -5.49260946e-03]
 [-5.08783273e-02  2.58860411e-03 -1.31703855e-04]
 [-1.09533870e+00  1.19976687e+00 -1.31415105e+00]
 [-5.71122587e-01  3.26180995e-01 -1.86289340e-01]
 [-1.98033720e-01  3.92173529e-02 -7.76635855e-03]
 [-1.04789364e+00  1.09808111e+00 -1.15067220e+00]
 [-9.16343391e-01  8.39685202e-01 -7.69439995e-01]
 [-1.33360016e+00  1.77848935e+00 -2.37179375e+00]
 [ 6.36083007e-01  4.04601604e-01  2.57360190e-01]
 [-3.32194835e-01  1.10353410e-01 -3.66588309e-02]
 [ 1.20635378e+00  1.45528948e+00  1.75559390e+00]
 [-5.00756741e-01  2.50757307e-01 -1.25568420e-01]
 [ 9.66754913e-01  9.34615076e-01  9.03543711e-01]
 [-4.62449878e-01  2.13859886e-01 -9.88994837e-02]
 [ 2.75164723e-01  7.57156238e-02  2.08342690e-02]
 [ 3.17283385e-02  1.00668741e-03  3.19405190e-05]
 [-2.26623344e+00  5.13581419e+00 -1.16389532e+01]
 [-1.05594598e-01  1.11502195e-02 -1.17740291e-03]
 [ 7.00263977e-01  4.90369648e-01  3.43388200e-01]
 [ 3.08013380e-01  9.48722437e-02  2.92219203e-02]
 [-5.78222096e-01  3.34340781e-01 -1.93323240e-01]
 [-2.06071541e-01  4.24654782e-02 -8.75092670e-03]
 [ 6.93279028e-01  4.80635822e-01  3.33214730e-01]
 [ 6.26973689e-01  3.93096000e-01  2.46460855e-01]
 [-3.58327329e-01  1.28398478e-01 -4.60086837e-02]
 [ 4.80031222e-01  2.30429977e-01  1.10613585e-01]
 [ 1.52610421e+00  2.32899404e+00  3.55428767e+00]
 [-1.28509682e-02  1.65147387e-04 -2.12230384e-06]
 [ 8.86878669e-01  7.86553800e-01  6.97577775e-01]
 [-3.11207384e-01  9.68500376e-02 -3.01404465e-02]
 [ 1.18732512e+00  1.40974092e+00  1.67382085e+00]
 [ 3.75856906e-01  1.41268417e-01  5.30967079e-02]
 [ 3.12301755e-01  9.75323841e-02  3.04595362e-02]
 [-4.38844830e-01  1.92584783e-01 -8.45148340e-02]
 [ 5.02259791e-01  2.52264887e-01  1.26702517e-01]
 [-6.35743678e-01  4.04170036e-01 -2.56948531e-01]
 [ 1.15118575e+00  1.32522869e+00  1.52558434e+00]
 [-1.61274180e-01  2.60093603e-02 -4.19463823e-03]
 [ 3.45989734e-01  1.19708896e-01  4.14180495e-02]
 [-5.44914305e-01  2.96931595e-01 -1.61802277e-01]
 [-5.50888956e-01  3.03478628e-01 -1.67183027e-01]
 [-5.30499160e-01  2.81429350e-01 -1.49298042e-01]
 [-4.11141425e-01  1.69037268e-01 -6.94982260e-02]
 [ 2.63828188e-01  6.96053132e-02  1.83638427e-02]
 [-6.45308495e-01  4.16423053e-01 -2.68721342e-01]
 [ 6.84398651e-01  4.68401521e-01  3.20573360e-01]
 [-4.89032641e-02  2.39152927e-03 -1.16953583e-04]
 [ 2.32864805e-02  5.42260183e-04  1.26273308e-05]
 [-1.60148084e-01  2.56474093e-02 -4.10738355e-03]
 [-7.80184686e-01  6.08688116e-01 -4.74889159e-01]
 [ 1.20582856e-01  1.45402253e-02  1.75330194e-03]
 [illegible]e-01  5.17428637e-01  3.72199416e-01]
```

Loading [MathJax]/jax/output/HTML-CSS/jax.js

```
 [ 1.63353443e-01  2.66843475e-02  4.35897987e-03]
 [ 4.62839425e-01  2.14220330e-01  9.91496146e-02]
 [-1.52416146e+00  2.32306814e+00 -3.54073095e+00]
 [ 9.52019542e-03  9.06341229e-05  8.62854563e-07]
 [ 2.34477043e+00  5.49794817e+00  1.28914270e+01]
 [ 1.44523215e+00  2.08869600e+00  3.01865053e+00]
 [ 1.67690086e+00  2.81199646e+00  4.71543932e+00]
 [-6.37291312e-01  4.06140208e-01 -2.58829623e-01]
 [ 4.83349413e-01  2.33626649e-01  1.12923309e-01]
 [-1.16144363e-02  1.34895134e-04 -1.56673093e-06]
 [ 9.08360124e-01  8.25118124e-01  7.49504387e-01]
 [ 1.22966623e+00  1.51207900e+00  1.85935259e+00]
 [ 2.43983641e-01  5.95280156e-02  1.45238619e-02]
 [ 1.52687168e+00  2.33133721e+00  3.55965257e+00]
 [ 1.83600307e+00  3.37090731e+00  6.18899584e+00]
 [ 9.47248638e-01  8.97279978e-01  8.49947214e-01]
 [ 4.70481455e-01  2.21352801e-01  1.04142390e-01]
 [ 1.41727889e+00  2.00867939e+00  2.84685898e+00]
 [ 1.05159092e+00  1.10584342e+00  1.16289496e+00]
 [-1.94384947e-01  3.77855077e-02 -7.34493416e-03]
 [ 3.28921765e-01  1.08189531e-01  3.55858915e-02]
 [ 1.86460972e+00  3.47676945e+00  6.48281813e+00]
 [-8.18179011e-01  6.69416904e-01 -5.47702849e-01]]
<NDArray 200x3 @cpu(0)>,
[[ 1.25765383e-01  1.58169325e-02  1.98922236e-03 ...  6.25876950e-08
   7.87136489e-09  9.89945237e-10]
 [-2.25531623e-01  5.08645140e-02 -1.14715565e-02 ...  6.69359679e-06
  -1.50961773e-06  3.40466556e-07]
 [-8.72416794e-02  7.61111081e-03 -6.64006046e-04 ...  3.35576966e-09
  -2.92762981e-10  2.55411334e-11]
 ...
 [ 3.28921765e-01  1.08189531e-01  3.55858915e-02 ...  1.37006413e-04
   4.50643929e-05  1.48226591e-05]
 [ 1.86460972e+00  3.47676945e+00  6.48281813e+00 ...  1.46117935e+02
   2.72452942e+02  5.08018402e+02]
 [-8.18179011e-01  6.69416904e-01 -5.47702849e-01 ...  2.00810626e-01
  -1.64299041e-01  1.34426028e-01]]
<NDArray 200x10 @cpu(0)>,
[   5.3321605     4.681967      4.892367    -55.73611       5.3499584
    2.5656848     6.2678165    -7.9106655     5.329318      4.965453
    8.770981      1.02188       5.542724      7.1657467    -8.829475
    0.5014815     6.882111      3.8610973    17.947128      5.8327003
    4.840865   -142.31438       6.1463094     3.6211767     4.8796453
   -6.7179675     4.998077     15.988336    116.26475    -11.93434
   -1.8376622     7.35403       5.240231    -29.101429     3.4611325
   57.107998    -2.4999328    -1.7207047     7.1990676     8.061852
    0.89318025    4.373194      6.7996335     6.825479      4.5852647
    6.5754905     5.3792925     4.9131813   -25.762821      5.0223403
    5.1912293    -2.4103968    39.75366       0.16852264    1.8409004
    4.950887    -93.70178       3.7843926    40.37843       4.475295
    5.3846307    10.262028    -1.8478428     5.0513177     1.3262482
    5.3185453    44.5604      -6.280218     10.5420475     4.971341
    5.3977866     5.228226      7.6699686     5.245119      4.6273413
  -19.352753    -2.5316908     4.7716637    -1.5653243     5.2358985
   12.720884     5.2102017   -12.520911      4.858167    -66.81847
    3.2701366     5.232396      3.4016063     6.9717617     7.4567
    0.598524     -4.4479046   -18.50684       5.050211    -21.737488
   -9.67158       4.1920223    20.997124     -7.819911      5.057172
    1.7525537   -35.193012      7.127098     13.394195    -0.8093322
   -5.448856     21.137714     47.872494      4.804509      5.331542
    1.7419873     4.92291       5.1017556     9.181443    -35.695152
                 33.71332     18.650951      7.426802      5.305392
```

```
 4.736012    -11.413436     5.7106915     5.839211      5.0817366
 4.519102      5.013267    -7.6337395     1.9551197     4.536426
-6.365321     -3.2750306  -15.86683       5.8586707     3.9689035
11.384986      2.9123888    7.9117794     3.1086183     5.1617584
 5.2009506   -80.23683      4.761033      5.8903365     5.119829
 2.0732548     4.63576      5.987012      5.6245914     3.9379733
 5.4050403    18.748072     4.849594      7.249881      4.155419
11.050908      5.3853803    5.211962      3.2097178     5.572529
 1.4364127    10.520961     4.5183334     5.3174205     2.4379816
 2.3892195     2.5062492    3.4029572     5.3407636     1.4507158
 6.1479197     5.0565996    5.286122      4.7685266    -0.87311417
 5.161984      6.13034      5.2435746     5.373977    -24.528723
 4.933191     61.43945     16.538763     23.862965      1.4002619
 5.190399      4.999614     7.4173074    11.72774       5.407167
18.771433     30.327845     7.8279157     5.3585706    15.82301
 9.139929      4.7880363    5.400265     31.534882     -1.4132127 ]
<NDArray 200 @cpu(0)>
```

In [38]:

```python
#定义作图函数，y轴为对数尺度
#x1值，y1值，x标注，y标注，x2值，y2值，图例，窗口大小
def semilogy(x_vals, y_vals, x_label, y_label, x2_vals=None, y2_vals=None,
             legend=None, figsize=(3.5, 2.5)):
    d2l.set_figsize(figsize)#设置窗口大小默认3.5*2.5
    d2l.plt.xlabel(x_label)#显示x轴意义
    d2l.plt.ylabel(y_label)#显示y轴意义
    d2l.plt.semilogy(x_vals, y_vals)#进行第一条曲线的绘制
    if x2_vals and y2_vals:#如果有第二条曲线，就要画第二条线，并显示图例
        d2l.plt.semilogy(x2_vals, y2_vals, linestyle=':')#进行第二条曲线的绘制
        d2l.plt.legend(legend)#显示图例
```

In [61]:

```python
#定义训练+显示一体化函数
#批次数，误差，
num_epochs, loss = 100, gloss.L2Loss()
#参数列表：训练输入，测试输入，训练输出，测试输出，学习率
def fit_and_plot(train_features, test_features, train_labels, test_labels,lear
nrate ):
    net = nn.Sequential()#一个有序的容器
    net.add(nn.Dense(1))
    net.initialize()
    batch_size = min(10, train_labels.shape[0])#批大小超过10个按10个记
    train_iter = gdata.DataLoader(gdata.ArrayDataset(
        train_features, train_labels), batch_size, shuffle=True)
    #训练器采用随机梯度下降SGD训练，学习率为learnrate做输入参数
    trainer = gluon.Trainer(net.collect_params(), 'sgd',{'learning_rate': lear
nrate})
    train_ls, test_ls = [], []#定义损失初值
    for _ in range(num_epochs):#不同训练批次
        for X, y in train_iter:#监督学习不同的特征与标签
            with autograd.record():
                l = loss(net(X), y)
            l.backward()
            trainer.step(batch_size)
        #训练，每一批后的损失值，并外挂到train_ls
        train_ls.append(loss(net(train_features),
                             train_labels).mean().asscalar())
        #测试，每一批后的损失值，并外挂到test_ls
        test_ls.append(loss(net(test_features),
                            test_labels).mean().asscalar())
    #最后的训练与测试误差
    print('final epoch: train loss', train_ls[-1], 'test loss', test_ls[-1])
    #　　　　练的结果，不同批次测试的结果，显示窗口
```

```
    semilogy(range(1, num_epochs + 1), train_ls, 'epochs', 'loss',
             range(1, num_epochs + 1), test_ls, ['train', 'test'])
    print('weight:', net[0].weight.data().asnumpy(),#输出训练后得出的系数（权重）
          '\nbias:', net[0].bias.data().asnumpy())#输出训练后得出的常数项（偏置）
```

2）先使用与数据生成函数同阶的三阶多项式函数拟合。

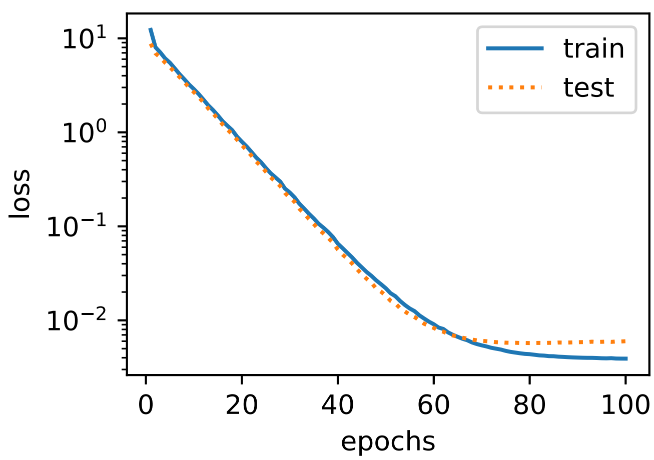
①打印出最佳参数的取值，与真实参数值进行对比，并对结果进行分析点评；

②尝试画出训练过程中模型分别在训练集和验证集上的学习曲线；观察分析这2条学习曲线，判断该模型是欠拟合，还是过拟合，还是表现很好？请说明理由。

③尝试使用不同的学习率η（四种情况：过大，过小，正好），画出不同的学习率η下，模型在训练集上的学习曲线；对比观察这三条学习曲线的走势，分析学习率对训练过程的影响。

In [40]:
```
#三阶拟合（模型复杂度刚刚好，学习率也刚刚好，最合适）学习率0.01
fit_and_plot(poly_features[:n_train, :], poly_features[n_train:, :],
             labels[:n_train], labels[n_train:], 0.01)
```

```
final epoch: train loss 0.003913792 test loss 0.0059956
weight: [[ 1.2012036 -3.4033813  5.6022115]]
bias: [5.0179133]
```

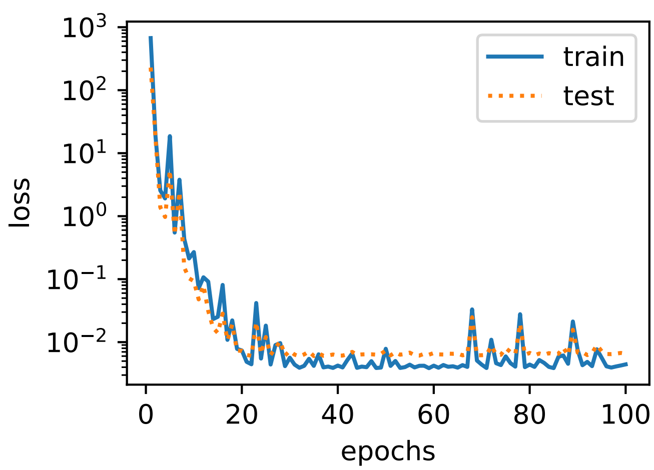


In [43]:
```
#三阶拟合（模型复杂度刚刚好，学习率过大导致发散）学习率0.1
fit_and_plot(poly_features[:n_train, :], poly_features[n_train:, :],
             labels[:n_train], labels[n_train:], 0.1)
```

```
final epoch: train loss 4.5542353e+34 test loss 1.6489693e+34
weight: [[-7.1200714e+16  6.7360635e+16 -4.5346428e+16]]
bias: [5.350943e+16]
```

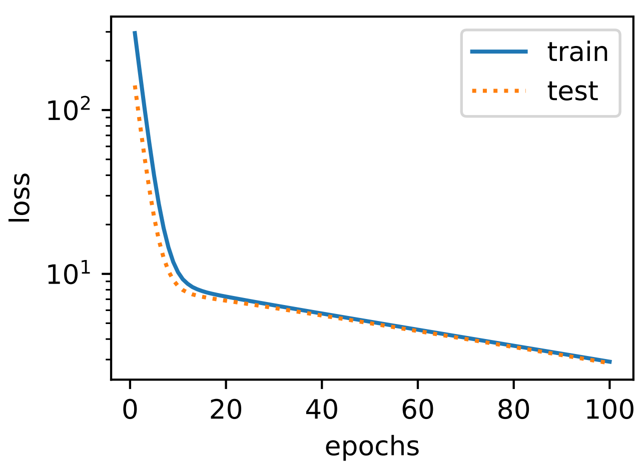
In [44]:
```python
#三阶拟合（模型复杂度刚刚好，学习率稍大学习曲线有点震荡）学习率0.05
fit_and_plot(poly_features[:n_train, :], poly_features[n_train:, :],
             labels[:n_train], labels[n_train:], 0.05)
```

```
final epoch: train loss 0.0044334345 test loss 0.006539675
weight: [[ 1.2146236 -3.4069202  5.606834 ]]
bias: [5.0215087]
```



In [12]:
```python
#三阶拟合（模型复杂度刚刚好，不过过小的学习率导致学习太慢了）学习率0.001
fit_and_plot(poly_features[:n_train, :], poly_features[n_train:, :],
             labels[:n_train], labels[n_train:], 0.001)
```

```
final epoch: train loss 2.9103785 test loss 2.8517141
weight: [[ 0.9256708 -2.3574617  5.750919 ]]
bias: [1.9527975]
```

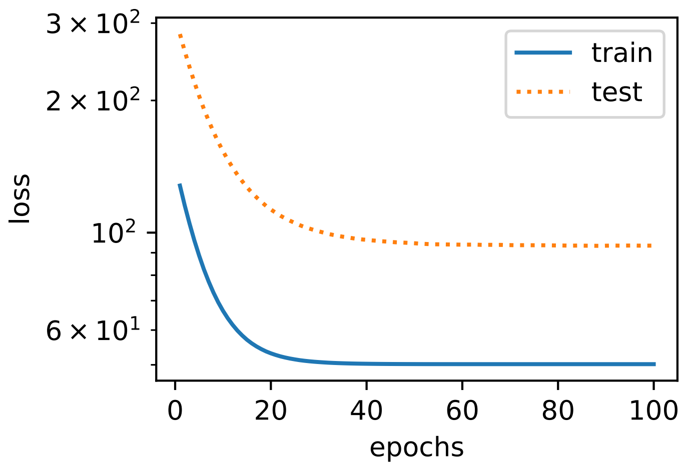


3）再试试线性函数拟合。

尝试画出训练过程中模型分别在训练集和验证集上的学习曲线；观察分析这2条学习曲线，判断该模型是欠拟合，还是过拟合，还是表现很好？请说明理由，并指出改进方案。

In [50]:
```
#线性拟合（模型太简单，会导致欠拟合）训练和测试的loss都很差
fit_and_plot(features[:n_train, :], features[n_train:, :], labels[:n_train],
             labels[n_train:],0.01)
```

```
final epoch: train loss 50.17723 test loss 93.4331
weight: [[14.462653]]
bias: [2.2241619]
```



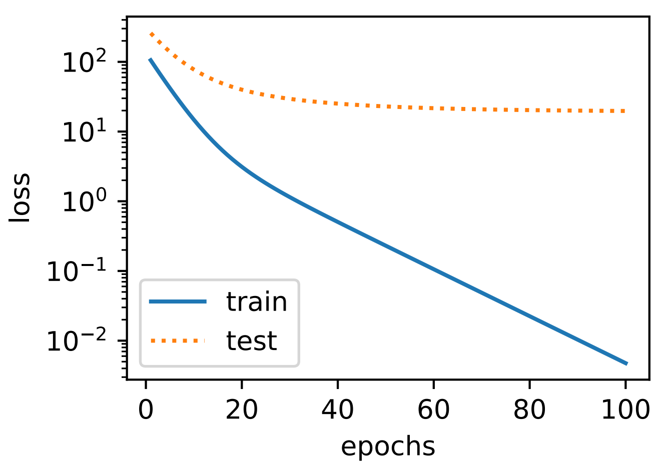
4）使用三阶多项式函数模型来拟合，请注意，这次，只使用2个样本来训练模型。

①打印出最佳参数的取值，并与真实参数值进行对比，并对结果进行分析点评。

②尝试画出训练过程中模型分别在训练集和验证集上的学习曲线；观察分析这2条学习曲线，判断该模型是欠拟合，还是过拟合，还是表现很好？请说明理由，并指出改进方案。

In [51]:
```
#三阶拟合，训练样本不足（欠拟合）需要更多的训练数据，后面的曲线应该是延伸的
fit_and_plot(poly_features[0:2, :], poly_features[n_train:, :], labels[0:2],
             labels[n_train:],0.01)
```

```
final epoch: train loss 0.004758147 test loss 19.740759
weight: [[ 2.4169416  -0.21812136  4.527212  ]]
bias: [-0.44129366]
```



5）使用10阶多项式函数模型来拟合。

Loading [MathJax]/jax/output/HTML-CSS/jax.js 的取值，并与真实参数值进行对比；

②尝试画出训练过程中模型分别在训练集和验证集上的学习曲线；观察分析这2条学习曲线，判断该模型是欠拟合，还是过拟合，还是表现很好？请说明理由，并指出改进方案。

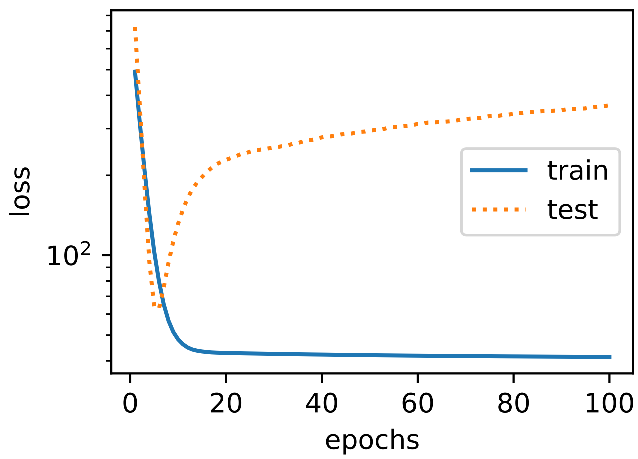
③尝试利用L2正则化技术，让该模型的表现好起来。画出训练过程中模型分别在训练集和验证集上的学习曲线；观察分析这2条学习曲线，判断该模型是欠拟合，还是过拟合，还是表现很好？并说明理由。

④打印出③中模型的最佳参数，并与①进行对比，尝试总结下L2正则化技术所带来的效果，并分析为何L2正则化技术可以解决过拟合问题。

⑤ 在③中使用不同的正则化强度，打印出不同正则化强度下的最佳参数，并对结果进行分析点评。

In [62]:
```
#十阶拟合（模型太复杂，会导致过拟合）
fit_and_plot(ten_features[:n_train, :],ten_features[n_train:, :], labels[:n_train],
             labels[n_train:],0.00000004)
```

```
final epoch: train loss 41.39038 test loss 367.50558
weight: [[-0.06136507  0.00243929 -0.0610089  -0.00351013  0.05785299 -0.04326928
   0.0338949  -0.00164474  0.04598321  0.00667603]]
bias: [6.871514e-05]
```



可以看出来最佳参数的取值，与真实参数值出入较大,而且有过拟和的现象，一般解决这种问题的方法如下

1.正则化方法，也就是权重衰减

2.加大训练数据

3.去掉神经网络的一部分进行训练，也就是丢弃法

4.提前终止

In [56]:
```
#进行L2正则化，定义惩罚项
def l2_penalty(w):
    return (w**2).sum() / 2
```

In [57]:
```
#加入了L2正则化的拟合函数
#批次数，误差，
```

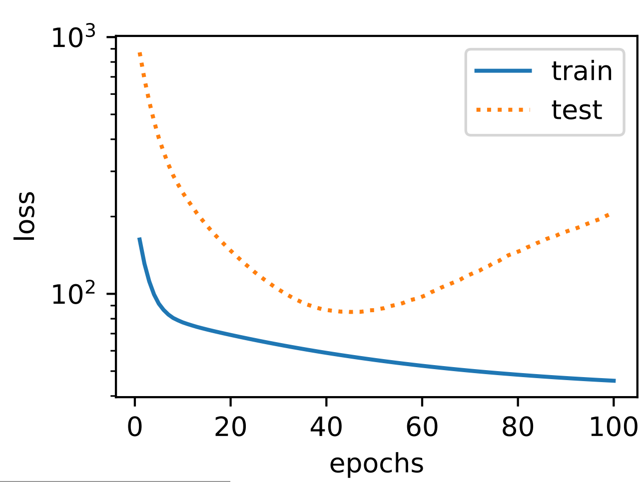
Loading [MathJax]/jax/output/HTML-CSS/jax.js

```
num_epochs, loss = 100, gloss.L2Loss()
#参数列表：训练输入，测试输入，训练输出，测试输出，学习率,惩罚项系数
def fit_and_plot_L2(train_features, test_features, train_labels, test_labels,learnrate ,lambd):
    net = nn.Sequential()#一个有序的容器
    net.add(nn.Dense(1))
    net.initialize()
    batch_size = min(10, train_labels.shape[0])#批大小超过10个按10个记
    train_iter = gdata.DataLoader(gdata.ArrayDataset(
        train_features, train_labels), batch_size, shuffle=True)
    #训练器采用随机梯度下降SGD训练，学习率为learnrate做输入参数
    trainer = gluon.Trainer(net.collect_params(), 'sgd',{'learning_rate': learnrate})
    train_ls, test_ls = [], []#定义损失初值
    for _ in range(num_epochs):#不同训练批次
        for X, y in train_iter:#监督学习不同的特征与标签
            with autograd.record():
                # 添加了L2范数惩罚项
                l = loss(net(X), y) + lambd * l2_penalty(net[0].weight.data().asnumpy())
            l.backward()
            trainer.step(batch_size)
        #训练，每一批后的损失值，并外挂到train_ls
        train_ls.append(loss(net(train_features),
                             train_labels).mean().asscalar())
        #测试，每一批后的损失值，并外挂到test_ls
        test_ls.append(loss(net(test_features),
                            test_labels).mean().asscalar())
    #最后的训练与测试误差
    print('final epoch: train loss', train_ls[-1], 'test loss', test_ls[-1])
    #不同批次训练的结果，不同批次测试的结果，显示窗口
    semilogy(range(1, num_epochs + 1), train_ls, 'epochs', 'loss',
             range(1, num_epochs + 1), test_ls, ['train', 'test'])
    print('weight:', net[0].weight.data().asnumpy(),#输出训练后得出的系数（权重）
          '\nbias:', net[0].bias.data().asnumpy())#输出训练后得出的常数项（偏置）
```

In [58]:

```
#十阶拟合（模型太复杂）加入了L2惩罚项，lambd=1，还是有过拟和
fit_and_plot_L2(ten_features[:n_train, :],ten_features[n_train:, :], labels[:n_train],
                labels[n_train:],0.00000004,1)
```

```
final epoch: train loss 45.867153 test loss 205.78508
weight: [[ 0.05440721  0.01373051 -0.00670504 -0.03651985  0.05950128  0.01235772
  -0.03963249 -0.04367105  0.04960401  0.0093249 ]]
bias: [8.3577805e-05]
```

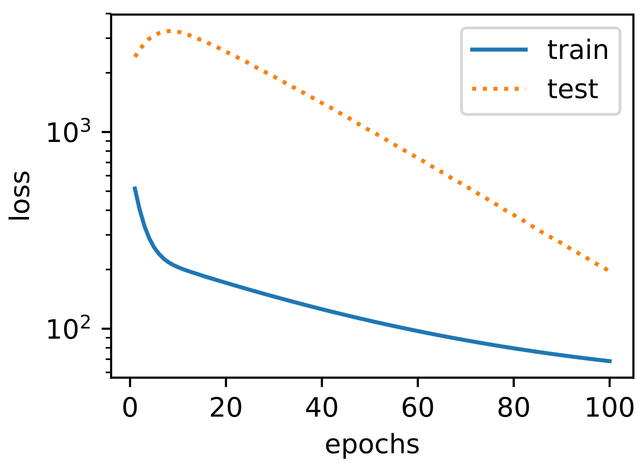


Loading [MathJax]/jax/output/HTML-CSS/jax.js

In [59]:

```
#十阶拟合（模型太复杂）加入了L2惩罚项，lambd=0.1，过拟和消失了
fit_and_plot_L2(ten_features[:n_train, :],ten_features[n_train:, :], labels[:n_train],
             labels[n_train:],0.00000004,0.1)
```

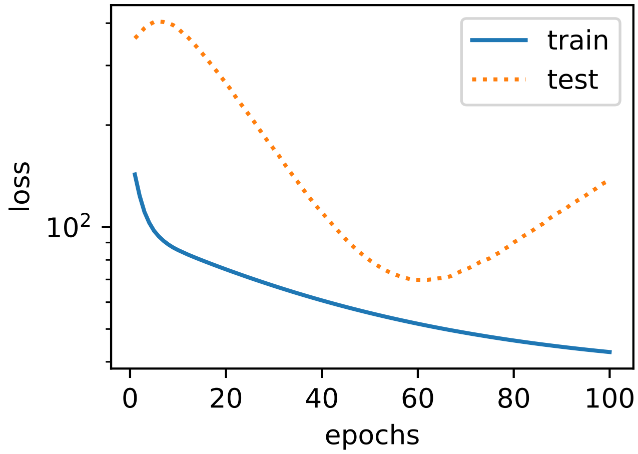
```
final epoch: train loss 68.418335 test loss 196.00613
weight: [[-0.00789979  0.0175912  -0.05790495 -0.01933222  0.03253733  0.05781509
  -0.01514692  0.05282685  0.02166412 -0.01926502]]
bias: [8.2431376e-05]
```



In [63]:

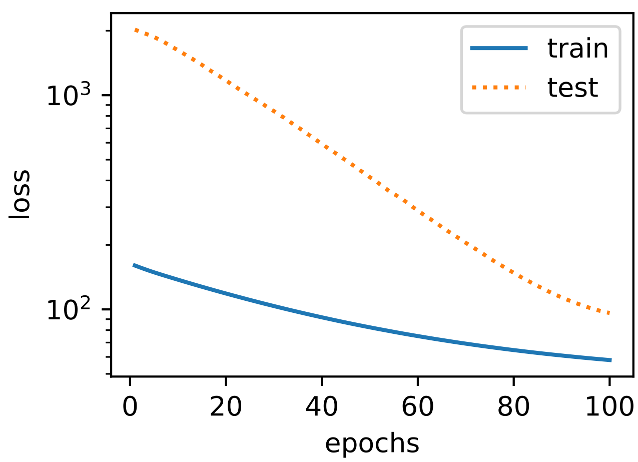
```
#十阶拟合（模型太复杂）加入了L2惩罚项，lambd=10，还是有过拟和，看来lambd不能太大
fit_and_plot_L2(ten_features[:n_train, :],ten_features[n_train:, :], labels[:n_train],
             labels[n_train:],0.00000004,10)
```

```
final epoch: train loss 42.74672 test loss 137.46866
weight: [[ 0.02450505  0.03014343  0.0049615   0.00328209 -0.02436037 -0.02484084
   0.0810871  -0.06879107  0.02687113  0.01341972]]
bias: [8.616818e-05]
```



In [64]:
```
#十阶拟合（模型太复杂）加入了L2惩罚项，lambd=0.01，调小了lambd更好一些了,目前是最好的情况
fit_and_plot_L2(ten_features[:n_train, :],ten_features[n_train:, :], labels[:n_train],
            labels[n_train:],0.00000004,0.01)
```

```
final epoch: train loss 58.013126 test loss 96.21356
weight: [[-0.00370966  0.02089359 -0.01593382 -0.06911948  0.07090588  0.06669808
  -0.03134783  0.02703604  0.03238817 -0.01092693]]
bias: [8.282152e-05]
```



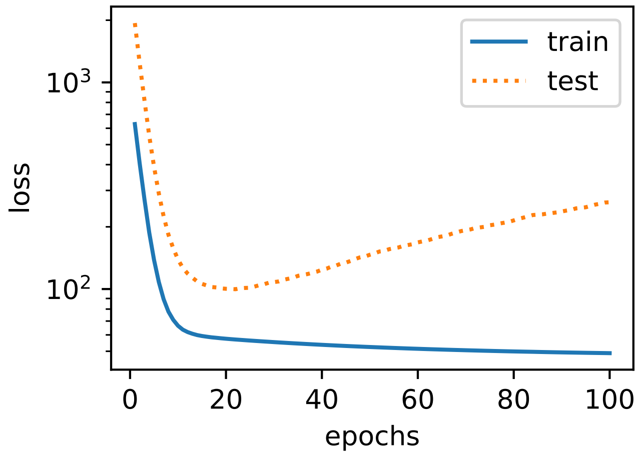
L1和L2正则都是比较常见和常用的正则化项，都可以达到防止过拟合的效果。L1正则化的解具有稀疏性，可用于特征选择。L2正则化的解都比较小，抗扰动能力强。在求解过程中，L2通常倾向让权值尽可能小，最后构造一个所有参数都比较小的模型。因为一般认为参数值小的模型比较简单，能适应不同的数据集，也在一定程度上避免了过拟合现象。参数足够小，数据偏移得多一点也不会对结果造成什么影响，可以说“抗扰动能力强”。

过拟合的时候，拟合函数的系数往往非常大，就是拟合函数需要顾忌每一个点，最终形成的拟合函数波动很大。在某些很小的区间里，函数值的变化很剧烈。这就意味着函数在某些小区间里的导数值（绝对值）非常大，由于自变量值可大可小，所以只有系数足够大，才能保证导数值很大。

而正则化是通过约束参数的范数使其不要太大，所以可以在一定程度上减少过拟合情况。

In [67]:
```
#十阶拟合（模型太复杂）加入了L2惩罚项，lambd=0.0001，继续调小了，又出现了过拟和，看来lambd在0.01的数量级比较合适
fit_and_plot_L2(ten_features[:n_train, :],ten_features[n_train:, :], labels[:n_train],
            labels[n_train:],0.00000004,0.0001)
```

```
final epoch: train loss 48.949074 test loss 263.03705
weight: [[ 0.0661952   0.01285097 -0.05049563 -0.0209086  -0.05381117  0.019
80051
  -0.01213619  0.01173303  0.05252288  0.00036133]]
bias: [4.4678276e-05]
```



总的来说还是要模型好最重要，用**10**阶去拟合的话，正则化的效果都不如用三阶去拟合