In [6]:
```
import cvxpy as cp
import numpy as np
import matplotlib.pyplot as plt
```

In [8]:
```
# 设置参数和生成原始向量
# 向量大小（人数）
n = 100
# 阳性数量
k = 5
# 随机种子
np.random.seed(1)
# 原始向量（n维，k稀疏）
x_orig = np.zeros(n)
S = np.random.randint(n, size=k)
x_orig[S] = 1

# 生成检测矩阵和结果向量
# 检测次数
m = 20
# 检测矩阵
A = np.random.randint(2, size=(m, n))
# 结果向量
b = A @ x_orig

# 构建和求解优化问题
# 优化变量
x = cp.Variable(n)
# 成本函数（L1 范数）
cost = cp.norm1(x)
# 约束（线性方程）
constraints = [A @ x == b]
# 优化问题
prob = cp.Problem(cp.Minimize(cost), constraints)
# 通过 CVXPY 求解
prob.solve()

# 打印结果
print("status:", prob.status)
print("optimal value:", prob.value)
print("original vector non-zero indices:", S)
print("recovered vector non-zero indices:", np.where(x.value > 0.5)[0])
```

```
status: optimal
optimal value: 5.000000001570281
original vector non-zero indices: [37 12 72  9 75]
recovered vector non-zero indices: [ 9 12 37 72 75]
```

In [3]:
```
print(np.nonzero(x_orig))
x_eat=np.round(x.value)
print(x_eat.nonzero())
```

```
(array([ 9, 12, 37, 72, 75], dtype=int64),)
(array([ 9, 12, 37, 72, 75], dtype=int64),)
```

In [10]:

```
plt.figure(figsize=(12, 5))

# 左图：原始稀疏向量
plt.subplot(1, 2, 1)
plt.stem(range(n), x_orig, basefmt=" ", use_line_collection=True)
plt.title(' $x_{orig}$')
plt.xlabel('index')
plt.ylabel('value')
plt.ylim(0, 1.2)
plt.grid(True, alpha=0.3)

# 右图：重构的向量
plt.subplot(1, 2, 2)
plt.stem(range(n), x.value, basefmt=" ", use_line_collection=True)
plt.title('vector')
plt.xlabel('index')
plt.ylabel('value')
plt.ylim(0, 1.2)
plt.grid(True, alpha=0.3)

plt.tight_layout()
plt.show()
```