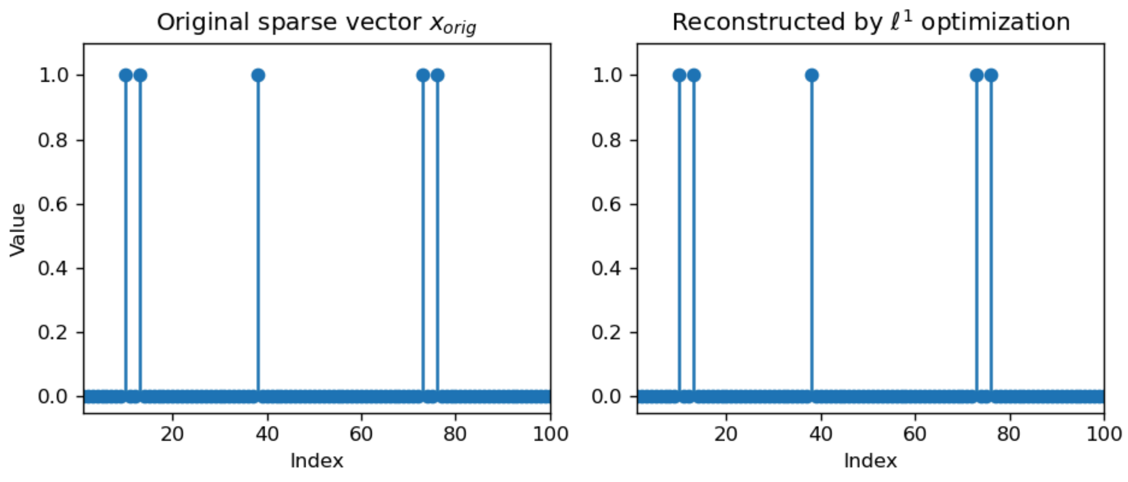
In [1]:

```
import cvxpy as cp
import numpy as np
import matplotlib.pyplot as plt
# 向量大小（人数）
n = 100
# 问题数量
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
# 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
# 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(np.nonzero(x_orig))
x_est=np.round(x.value)
print(x_est.nonzero())
# ===== 画图：原始稀疏向量 vs 通过 l1 优化重构的向量 =====
x_rec = x_est  # 也可以换成 x.value（不取整），看连续解

idx = np.arange(1, n + 1)  # 横轴从 1 到 n，更贴近教材示意

fig, axes = plt.subplots(1, 2, figsize=(8, 3.5), dpi=120)

# Left: original sparse vector
ml, sl, bl = axes[0].stem(idx, x_orig, use_line_collection=True)
bl.set_visible(False)
axes[0].set_title("Original sparse vector $x_{orig}$")
axes[0].set_xlabel("Index")
axes[0].set_ylabel("Value")
axes[0].set_xlim(1, n)
axes[0].set_ylim(-0.05, 1.1)

# Right: reconstructed by l1 optimization
ml2, sl2, bl2 = axes[1].stem(idx, x_rec, use_line_collection=True)
bl2.set_visible(False)
axes[1].set_title("Reconstructed by $\\ell^1$ optimization")
```

```
axes[1].set_xlabel("Index")
axes[1].set_xlim(1, n)
axes[1].set_ylim(-0.05, 1.1)

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

```
status: optimal
optimal value: 5.000000001570281
(array([ 9, 12, 37, 72, 75], dtype=int64),)
(array([ 9, 12, 37, 72, 75], dtype=int64),)
```


Original sparse vector x_{orig}
Reconstructed by ℓ^1 optimization
Value
Index
Index