

NUMERICAL RECIPES

Webnote No. 13, Rev. 1

Interface to AMD and LDL Packages

In the main text we gave the interface NR1ddl.h to the required external packages AMD and LDL. Here is the corresponding implementation, NR1ddl.cpp.

```
NR1ddl::NR1ddl(NRsparseMat &adat) : n(adat.ncols), nz(adat.nvals), interior.h
    Ap(&adat.col_ptr[0]), Ai(&adat.row_ind[0]), Ax(&adat.val[0]),
    PP(n), PPinv(n), PPattern(n), LLnz(n), LLp(n+1), PParent(n), FFlag(n),
    YY(n), DD(n), Y(&YY[0]), D(&DD[0]), P(&PP[0]), Pinv(&PPinv[0]),
    Pattern(&PPattern[0]), Lnz(&LLnz[0]), Lp(&LLp[0]), Parent(&PParent[0]),
    Flag(&FFlag[0]) {}

void NR1ddl::order() {
    if (amd_order(n, Ap, Ai, P, (Doub *) NULL, Info) != AMD_OK)
        throw("call to AMD failed");
    amd_control((Doub *) NULL);
    //amd_info(Info);
    ldl_symbolic(n, Ap, Ai, Lp, Parent, Lnz, Flag, P, Pinv);
    lnz = Lp[n];
    /* find # of nonzeros in L, and flop count for ldl_numeric */
    Doub flops = 0;
    for (Int j = 0; j < n; j++)
        flops += ((Doub) Lnz[j]) * (Lnz[j] + 2);
    cout << "Nz in L: " << lnz << " Flop count: " << flops << endl;
    /* ----- */
    /* allocate remainder of L, of size lnz */
    /* ----- */
    LLi=new VecInt(lnz);
    LLx=new VecDoub(lnz);
    Li=&(*LLi)[0];
    Lx=&(*LLx)[0];
}

void NR1ddl::factorize() {
    /* ----- */
    /* numeric factorization to get Li, Lx, and D */
    /* ----- */
    Int dd = ldl_numeric(n, Ap, Ai, Ax, Lp, Parent, Lnz, Li, Lx, D,
        Y, Flag, Pattern, P, Pinv);
    if (dd != n)
        throw("Factorization failed since diagonal is zero.");
}

void NR1ddl::solve(VecDoub_0 &y, VecDoub &rhs) {
    B=&rhs[0];
    X=&y[0];
    /* solve Ax=b */
    /* the factorization is LDL' = PAP' */
    ldl_perm(n, Y, B, P); /* y = Pb */
    ldl_lsolve(n, Y, Lp, Li, Lx); /* y = L\y */
    ldl_dsolve(n, Y, D); /* y = D\y */
}
```

```
    ldl_ltsolve (n, Y, Lp, Li, Lx) ;    /* y = L'\y */
    ldl_permt (n, X, Y, P) ;          /* x = P'y */
}

NR1dl::~NR1dl() {
    delete LLx;
    delete LLi;
}
```