

## Help

```

/*
 * File written by Jérôme Lelong <jerome.lelong@gmail.com>
 * for Premia release 11
 * February 2009
 */

#include <stdlib.h>
#include "pnl/pnl_mathtools.h"
#include "pnl/pnl_random.h"
#include "pnl/pnl_matrix.h"
#include "pnl/pnl_vector.h"

#include "bsnd_path.h"

/**
 * Draws one path of a multidimensional Black-Scholes
 * model for a given set of
 * Brownian increments. The asset path may be drifted by ad
 * ding a linear drift
 * to the underlying Brownian motion
 *
 * @param path the matrix containing the path on exit
 * @param mod a PremiaBSnd structure describing the B&S
 * parameters
 * @param G a matrix of size timesteps x dim containing i.i.
 * d. samples
 * from the normal law used to build the Brownian increments
 * @param T the maturity time
 * @param drift a vector of drift If NULL, it means do not
 * drift the path
 */
void premia_bs_path (PnlMat *path, const PremiaBSnd *mod ,
                    const PnlMat *G,
                        double T, const PnlVect *drift)
{
    int i, j, k, d, timesteps;
    double b, bdt, sigma_j, St, Lj_mu, dt, sqrt_dt, wt;
    d = mod->d;
    timesteps = mod->timesteps;

```

```

dt = T / timesteps;
sqrt_dt = sqrt (dt);
pnl_mat_resize (path, timesteps+1, d);
pnl_mat_set_row (path, mod->spot, 0);

for (j=0; j<d; j++)
{

    St = pnl_vect_get (mod->spot, j);
    /* compute L_j * mu */
    Lj_mu= 0.;
    if (drift != NULL)
    {
        for (k=0; k<=j; k++)
        {
            Lj_mu += pnl_mat_get(mod->LGamma, j, k)
                * pnl_vect_get (drift, k);
        }
    }
    sigma_j = pnl_vect_get (mod->sigma, j);
    b = mod->r - pnl_vect_get (mod->divid, j) - sigma_j *
sigma_j / 2.
    + sigma_j * Lj_mu;
    bdt = b * dt;
    for (i=0; i<timesteps; i++)
    {
        wt = 0.;
        for (k=0; k<=j; k++)
        {
            wt += pnl_mat_get(mod->LGamma, j, k) * pnl_
mat_get (G, i, k);
        }
        St *= exp (bdt + sigma_j * sqrt_dt * wt);
        pnl_mat_set (path, i+1, j, St);
    }
}
}

```

## References