

[Help](#)

```
#if defined(PremiaCurrentVersion) && PremiaCurrentVersion < (2007+2) //The "#els
#else

#include <stdlib.h>
#include "
href../../mod/hk1d/hk1d_stdi/currentzcb_h_src.pdfcurrentzcb.h"
#include "
href../../common/optype_h_src.pdfoptype.h"

/* defined in premia_obj.c */
extern char premia_data_dir[MAX_PATH_LEN];
extern char *path_sep;

static FILE *Entrees;                                /* File variable of the code */
static double *tm;                                    /* Times T for which P(0,T) is read in t
static double *Pm;                                    /* Values of P(0,tm) read in the file */
static int Nvalue;                                    /* Number of values read for Pm */

int lecturehk(char *init)
{
    int i;
    char ligne[20];
    char *pligne;
    double p, tt;

    Entrees = fopen(init, "r");

    if (Entrees == NULL)
    {
        printf("LE FICHER N'A PU ETRE OUVERT. VERIFIER LE CHEMIN\ n");
    }
    else {}

    i = 0;
    pligne = ligne;
```

```

Pm = malloc(100 * sizeof(double));
tm = malloc(100 * sizeof(double));

while (1)
{
    pligne = fgets(ligne, sizeof(ligne), Entrees);
    if (pligne == NULL) break;
    else
    {
        sscanf(ligne, "%lf t=%lf", &p, &tt);

        Pm[i] = p;
        tm[i] = tt;
        i++;
    }

}

fclose(Entrees);

return i;
}

double bond(double T, double FM)
{
    double POT;
    int i = 0;

    if (T > 0)
    {
        if (FM > 0)
        {
            POT = exp(-FM * T);
        }
        else
        {
            while (tm[i] < T && i < Nvalue)
            {

```

```

        i = i + 1;
    }

    if (i == 0)
    {
        POT = 1 * (1 - T / tm[0]) + Pm[0] * (T / tm[0]);
    }
    else
    {
        if (i < Nvalue)
        {
            POT = Pm[i - 1] * (tm[i] - T) / (tm[i] - tm[i - 1]) + Pm[i] *
        }
        else
        {
            POT = Pm[i - 1] + (T - tm[i - 1]) * (Pm[i - 1] - Pm[i - 2]) /
        }
    }
}
else
{
    POT = 1;
}
return POT;
}

```

```

double CurrentZCB(double T, int flat_flag, double r_flat, char *init)
{
    if (flat_flag == 0) return exp(-r_flat * T);

    Nvalue = lecturehk(init);
    if (T > tm[Nvalue - 1])
    {
        printf("\ nError : P(0,T) can be deduced from datas on file only for T<=%f\n", T);
        printf("But here T=%f !!\ n", T);
        return -1;
    }

    return bond(T, -1);
}

```

```
}  
#endif //PremiaCurrentVersion
```