Bond Duration & Convexity on the HP-12C

Tony Hutchins, #1049

Example 1: 10 annual coupons. 5% yield. 4% coupon. 10 years to maturity.

10 n 5 i 4 PMT 0 STO 0 R/S →7.9615135(duration) x≷y →78.29424(convexity).

Keystrokes	Display	Keystrokes	Display	Keystrokes	Display
f P/R		×	20- 20	−	41- 30
f CLEAR PRGM	00-	RCL FV	21-45 15	g LSTx	42-43 36
RCL i	01-45 12	−	22- 30	×	43- 20
RCL i	02-45 12	RCL n	23-45 11	−	44- 30
RCL PMT	03-45 14	×	24- 20	STO 2	45-44 2
−	04- 30	FV	25- 15	RCL 1	46-45 1
RCL n	05-45 11	PV	26- 13	+	47- 40
×	06- 20	−	27- 30	EEX	48- 26
g BEG	07-43 7	%T	28- 23	RCL i	49-45 12
FV	08- 15	EEX	29- 26	%	50- 25
PV	09- 13	2	30- 2	+	51- 40
%T	10- 23	g END	31-43 8	÷	52- 10
STO 1	11-44 1	FV	32- 15	g LSTx	53-43 36
2	12- 2	PV	33- 13	÷	54- 10
RCL i	13-45 12	STO ÷ 1	34-44 10 1	RCL 1	55-45 1
%	14- 25	÷	35- 10	g LSTx	56-43 36
+	15- 40	RCL 1	36-45 1	÷	57- 10
×	16- 20	ENTER	37- 36	g GTO 00	58-43,33 00
g LSTx	17-43 36	+	38- 40	f P/R	
EEX	18- 26	RCL 0	39-45 0	12c platinum needs 6 extra steps	
2	19- 2	STO − 1	40-44 30 1		

	R_0	n	i	PV	PMT	FV
Input	f=accrual	#coupons	yield%	n/a	coupon%	n/a
Output	unchanged	unchanged	unchanged	-price	unchanged	100

i and PMT are per coupon period.

Register Usage			Calculation	Lines	Calculation	Lines
R_0=f	R_1=Df	R_2=Cf	D and C	1-35	Df=D-f	40
output X	eDf		Cf=C-f·(2·D-f)	36-45	1+i	48-51
output Y	eCf		$eCf=(Cf+Df)/(1+i)^2$	46-56	eDf=Df/(1+i)	55-57

Df & eDf are in coupon periods. Cf & eCf are in $(\text{coupon periods})^2$.

The first 35 steps do the tricky work. The rest can almost be done manually. Each time is weighted by the present value of the bond cashflow at that time, divided by the price. The weights add to 1. D is then just the weighted mean time (first moment of time, statistically). If the dirty price is calculated using compound interest and the pricing point is shifted forward by f then D becomes Df=D-f. C is the weighted mean time-squared (second moment of time, statistically). The financial markets quote the numbers eDf and eCf as shown in the previous table, because they are used to determine sensitivity in price with respect to a change in i, the **effective yield** per period. If the **continuous yield** were used instead then Df and Cf would suffice. These solver formulae use the "PV" function only available in the 200LX and 19B and 19BII. They show the 3 PVs as calculated by the 12C program, in lines 9, 26 and 33. g% is the coupon% stored in PMT.

{D=PV(n,i%,g%,n*(i%-g%),1,1)/PV(n,i%,g%,100,1,0)/i%*100}

{C=(PV(n,i%,g%,n*(i%-g%),1,1)*(200/i%+2)

-PV(n,i%,g%,n*((200+2*i%)-n*(i%-g%)),1,1))

/PV(n,i%,g%,100,1,0)/i%*100}

These closed formulae can also be calculated using summation loops (see the formulae later for a_n, a1 and a2), but on the 12C this takes more code and execution time then depends on n, but then the program does work for i=0, where D=n(1+g(n+1)/2)/(1+g·n) and C=n^2(1+g(n+1)(2n+1)/6/n)/(1+g·n). Here i% less than 0.01% is not recommended. Note how we change payment mode within the program (lines 7 and 31) - a very useful feature, peculiar to the 12C. The 38C requires a little extra code. The program can relatively easily be extended to use a face value of other than 100.

Example 2: Taken from page 77 of the HP-12C Owner's Handbook ("manual"). 29 semi-annual coupons. 8.25% yield. 6.75% coupon. Accrual=145/182.

29[n] 8.25[ENTER]2[÷][i] 6.75[ENTER]2[÷][PMT] 145[ENTER]182[÷][STO]0 [R/S] →16.6875 2[÷]→8.342873. [x≷y]4[÷]→98.18111. Set 'C' with [STO][EEX]. [RCL]0[n] [CLx][PMT] [FV]→90.31067 (correct dirty price from the manual).

This one just does D. It uses no storage registers and preserves 2 input stack levels. 10[n] 5[i] 4[PMT] [R/S] →8.3595892(D). Then 1.05 [÷]→ 7.9615135(eD)

Keystrokes	Display	Keystrokes	Display	Keystrokes	Display
[f][P/R]		[FV]	07- 15	[PV]	15- 13
[f] CLEAR [PRGM]	00-	[PV]	08- 13	[÷]	16- 10
[RCL][i]	01-45 12	[EEX]	09- 26	[RCL][i]	17-45 12
[RCL][PMT]	02-45 14	2	10- 2	[÷]	18- 10
[−]	03- 30	[X]	11- 20	[g][GTO]00	19-43,33 00
[RCL][n]	04-45 11	[g][LSTx]	12-43 36	[f][P/R]	
[X]	05- 20	[g][END]	13-43 8	12c platinum needs 4 extra steps	
[g][BEG]	06-43 7	[FV]	14- 15		

The following program is useful for finding the accrual given the settlement date (SETT), the last and next coupon dates (LCD and NCD). NCD[ENTER] SETT[ENTER] LCD[g][GTO]65[R/S] puts the actual/actual accrual fraction in R_0, and [g][GTO]59 [R/S] does the 30/360 accrual. E.g., for this example: [g][M.DY] 6.041982[ENTER]

4.281982[ENTER] 12.041981 [g][GTO]65[R/S]→0.70670. Note that R_3-R_6 are available for storing dates, if necessary.

Keystrokes	Display	Keystrokes	Display	Keystrokes	Display
[g][ΔDYS]	59-43 26	[g][ΔDYS]	65-43 26	[x≷y]	71- 34
[R↓]	60- 33	[x≷y]	66- 34	[R↓]	72- 33
[x≷y]	61- 34	[R↓]	67- 33	[÷]	73- 10
[g][LSTx]	62-43 36	[x≷y]	68- 34	[STO]0	74-44 0
[g][ΔDYS]	63-43 26	[g][LSTx]	69-43 36	[g][GTO]00	75-43,33 00
[g][GTO]72	64-43,33 72	[g][ΔDYS]	70-43 26		

The next program interfaces directly with the built-in bond program to produce the duration and convexity of a bond. From page 77 of the manual: 8.25[i] 6.75[PMT] [g][M.DY] 4.281982[ENTER] 6.041996 [f][PRICE] then [g][GTO]76[R/S]→16.6875, as above. Note how the program first stores 1-n in R_0. This is possible because the built-in bond programs (i.e. [f][PRICE]or[f][YTM]) leave 1-f in [n], an undocumented feature, till now<G>. They also leave 100+CPN/2 in [FV] which is also used here. The PV is the clean price, essential here where we are calculating n. The n is calculated using payment mode END and the 12C rounds this up so we automatically get the total number of coupons. Quite useful!

Keystrokes	Display	Keystrokes	Display	Keystrokes	Display
[EEX]	76- 26	[i]	83- 12	[÷]	90- 10
[RCL][n]	77-45 11	[RCL][PV]	84-45 13	[PMT]	91- 14
[−]	78- 30	[CHS]	85- 16	[−]	92- 30
[STO]0	79-44 0	[PV]	86- 13	[FV]	93- 15
[RCL][i]	80-45 12	[RCL][FV]	87-45 15	[g][END]	94-43 8
2	81- 2	[RCL][PMT]	88-45 14	[n]	95- 11
[÷]	82- 10	2	89- 2	[g][GTO]01	96-43,33 01

After running the above program the original bond can easily be manipulated using the built-in TVM itself. For example, if it were a German Moosmüller bond where simple interest is used in the odd period, what would its price be? We need to set [BEG] mode, put a coupon back with FV([RCL][FV][RCL][PMT][+][FV]), use n-f instead of n ([RCL][n] [RCL]0[−][n]) and clear 'C' with [STO][EEX], then [PV]→ -90.29863, the new dirty price - with 'C', [PV]→ -90.31067, as before. We can now also solve for i if we wish. At this moment in time<G> we seem to have replaced the built-in bond programs :-) We can even do an ex-div valuation on this bond - skip the next coupon by simply doing [g][END] 100[FV][PV]. The accrued interest is now f·PMT-PMT, not the usual f·PMT. In DataFile V22N1P32 I wrote there is no accrued interest for the ex-div case. It's negative. Also I see on P31 I have an 8.343873 which should be 8.342873. Finally, a challenge: Let $v=1/(1+i)$, $a_n=v+v^2+v^3+...+v^n=(1-v^n)/i$, $a1=v+2v^2+3v^3+...+n\cdot v^n$ & $a2=v+4v^2+9v^3+...+n^2v^n$, then given $PV+PMT\cdot a_n+FV\cdot v^n=0$([END]mode), $PV+PMT\cdot(1+i)\cdot a_n+FV\cdot v^n=0$ ([BEG]mode), $P=g\cdot a_n+v^n$, $D=(g\cdot a1+n\cdot v^n)/P$, & $C=(g\cdot a2+n^2v^n)/P$, derive the formulae used here :-) *Hint1*: first show that $a1=(a_n\cdot(1+i) - n\cdot v^n)/i$ and $a2=(2\cdot(1+i)\cdot a1 - a_n\cdot(1+i) - n^2v^n)/i$. *Hint2*: $v\cdot a1=v^2+2v^3+3v^4+...+n\cdot v^{n+1}$, so $a1-v\cdot a1=a1\cdot(1-v)=a1\cdot i/(1+i)= v+v^2+v^3+ ...+v^n$

- $n \cdot v^{n+1} = a_n - n \cdot v^{n+1}$. There is still plenty of manipulation required to get the formula for C as used here.

There is a third way, albeit more approximate, to get eDf and eCf directly. Before going on to that, I should point out that statistically the standard deviation, s, of the bond payment times is just the square root of the variance, $s^2=C-D^2$. This number is independent of "f", it is just a property of the bond payments stream, and the yield, but **not when** it is actually valued. One can even put f=D itself and s is unchanged, but Df=0. So, $s^2=C-D^2=Cf-Df^2$. For a normal bond $C>=D^2$ and s is real. For the next method I drop the "f" notation and derive eD and eC. To get back to the statistical D and C: $D=eD \cdot (1+i)$ and $C=eC \cdot (1+i)^2-D$. eD is sometimes called the "modified duration" or "volatility".

By Taylor Expansion...

$P(i+\Delta i)=P(i)+ \Delta i \cdot (dP/di) +(\Delta i)^2/2 \cdot (d^2P/di^2)+(\Delta i)^3/6 \cdot (d^3P/di^3)+ (\Delta i)^4/24 \cdot (d^4P/di^4) + ...$

Let $\Delta P= P(i+\Delta i)-P(i)$, and $P=P(i)$, then

$\Delta P/P=\Delta i \cdot (dP/di)/P +(\Delta i)^2/2 \cdot (d^2P/di^2)/P+(\Delta i)^3/6 \cdot (d^3P/di^3)/P+ (\Delta i)^4/24 \cdot (d^4P/di^4)/P + ...$

$= -\Delta i \cdot eD +(\Delta i)^2/2 \cdot eC -(\Delta i)^3/6 \cdot B + (\Delta i)^4/24 \cdot A + ...$

$eD=-(dP/di)/P$, and $eC=(d^2P/di^2)/P$, are the same as the eDf and eCf used earlier. In fact the approximate eD as derived below is often called the "effective" duration.

$B=-(d^3P/di^3)/P$ & $A=(d^4P/di^4)/P$, are the Butterfly and the Ant, which have tiny effects on $\Delta P/P$. Consider two $\Delta P/P$ denoted CH1 and CH2 caused by $\Delta i=+h$ and $\Delta i=-h$. Then

$CH1= -eD \cdot h+eC/2 \cdot h^2 - B/6 \cdot h^3+A/24 \cdot h^4 +...$,& $CH2=eD \cdot h+eC/2 \cdot h^2 +B/6 \cdot h^3 + A/24 \cdot h^4 +...$

$CH2-CH1=2 \cdot eD \cdot h + B/3 \cdot h^3 +..$, & $CH1+CH2= eC \cdot h^2 + A/12 \cdot h^4 +...$

Hence if we ignore A and B and higher terms we can calculate P(i), P(i+h) and P(i-h) and then find $eD \cong (CH2-CH1)/2h$ & $eC \cong (CH1+CH2)/h^2$. The markets use h=.01%=.0001=1BP=1 basis point= "an 01". The following little program does the job, using h=.01%. It requires P(i) stored in R_0, P(i+h) in R_1 and P(i-h) in R_2. Repeating the first example:

[g][END] 10[n] 5[i] 4[PMT] 100[FV][PV][STO]0 5.01[i][PV] [STO]1 4.99[i][PV] [STO]2

[R/S]→7.961514550 [x≷y]→78.29580000, c.f. 78.29424, so some significance has been lost, but the results are pretty good! eD is high by .000001 and eC by .002 - on the 12c platinum eC is high by only .00004.

Keystrokes	Display	Keystrokes	Display	Keystrokes	Display
[f][P/R]		[Δ%]	07- 24	[RCL]1	15-45 1
[f]CLEAR[PRGM]	00-	[STO]1	08-44 1	[−]	16- 30
[RCL]0	01-45 0	[RCL]2	09-45 2	5	17- 5
[RCL]2	02-45 2	[+]	10- 40	0	18- 0
[Δ%]	03- 24	[EEX]	11- 26	[×]	19- 20
[STO]2	04-44 2	6	12- 6	[g][GTO]00	20-43,33 00
[R↓]	05- 33	[×]	13- 20	[f][P/R]	
[RCL]1	06-45 1	[RCL]2	14-45 2		

This also works at i=0!! 0[i][PV][STO]0 .01[i][PV][STO]1 .01[CHS][i][PV] [STO]2 [R/S]→ 7.75 [x≷y]→ 77.0, both exactly correct! To verify the 77, remember eC=C+D for i=0 and here, using the formula given earlier: $C=n^2(1+g(n+1)(2n+1)/6/n)/(1+g\cdot n)$ =100*(1+.1*11*21/6/10)/(1+.1*10)=69.25. On the 12cp we get the same convexity but the duration shows as 7.750001450 - that "145" is a trace of the Butterfly (B above). The Ant (A) has got lost in the machine :-)

Bond Interface for Taylor ...

The following addition to the above gives us a full interface with the built-in bond application. At the end, the settlement date is in R_3 and the maturity date in R_4. Note how these dates are preserved in the stack, enabling re-runs for different prices. Instead of pressing [f][PRICE] just press [g][GTO]21 [R/S] as shown in this re-run of our bond example:

8.25[i] 6.75[PMT] [g][M.DY] 4.281982[ENTER]6.041996 [g][GTO]21 [R/S]→ 8.342874930 [x≷y]→ 98.18330000. eD is high by .000002 and eC by .002 (on 12cp by .0001).

The dollar value of 1BP changes can be recovered as follows:

DV+01: [RCL]0 [RCL]1 [%] → -.075301, DV-01: [RCL]0 [RCL]2 [%] → +.075389, and [RCL]0 → 90.31067, the correct dirty price, and [RCL][PV] →87.62180, the clean price.

Keystrokes	Display	Keystrokes	Display	Keystrokes	Display
[RCL][i]	21-45 12	[+]	32- 40	[i]	43- 12
[STO]3	22-44 3	[STO]1	33-44 1	[R↓]	44- 33
[STO]4	23-44 4	[R↓]	34- 33	[f][PRICE]	45-42 21
[EEX]	24- 26	[RCL]3	35-45 3	[+]	46- 40
2	25- 2	[i]	36- 12	[STO]0	47-44 0
[CHS]	26- 16	[R↓]	37- 33	[R↓]	48- 33
[STO][−]3	27-44 30 3	[f][PRICE]	38-42 21	[STO]4	49-44 4
[+]	28- 40	[+]	39- 40	[R↓]	50- 33
[i]	29- 12	[STO]2	40-44 2	[STO]3	51-44 3
[R↓]	30- 33	[R↓]	41- 33	[g][GTO]01	52-43,33
[f][PRICE]	31-42 21	[RCL]4	42-45 4	[f][P/R]	

The one problem with this is that it takes about 25 seconds to run. eD is pretty accurate but eC doesn't fare so well. But it is a surprisingly good practical solution!

Speed isn't a problem on the 12cp. As noted, the first two programs require modification to run on the newer 12cp. Each [FV][PV] sequence needs to be [FV][R↓][PV][PV].

Still room for a further hint on the previous challenge:

Hint3: $v\cdot a2=v^2+4v^3+9v^4+...+n^2\cdot v^{n+1}$, so $a2-v\cdot a2=a2\cdot(1-v)=a2\cdot i/(1+i)= v+3v^2+5v^3+ ...+(2n-1)v^n - n^2\cdot v^{n+1} = 2a1 - a_n - n^2\cdot v^{n+1}$. We are now well on the way. The resulting formulae, written using the classic HP TVM notation are really quite simple. For example, compare them with the closed formulae you'll find by googling "closed duration wikipedia" and "closed convexity wikipedia":-)