**Partial Fraction Expansion for HP15c CE**

**Pepin Torres, P.E.** pepin[dot]torres[at]gmail[dot]com

**Introduction:**

This software pac calculates the **partial fraction expansion of a transfer function** as implemented by Wayne Scott in the program **PF** in the **POLY** folder of the classic software package [HP 48 Goodies Disk, Vol. 1](https://www.hpcalc.org/details/232)

**Who is this for?**

This software pac is for anyone in dire need of performing a partial fraction expansion (I am looking at you Control Systems students). You provide the tranfer function, and this program gives you the expansion. Since the program is (almost) a direct rehosting of the original from HP48 to HP15c, I’ll just paste the original description which is clear and succinct enough.

PF will do a Partial Fraction expansion on a transfer function.

Example:

s + 5 1/18 5/270 2/3 1/9 2/27

----------------- = ----- + ----- - ------- - ------- - -----

(s-4)(s+2)(s-1)^3 (s-4) (s+2) (s-1)^3 (s-1)^2 (s-1)

2: { 1 5 }

1: { 4 -2 1 1 1 }

PF

1: { 5.5555e-2 1.85185e-2 -.6666 -.11111 -.074074 }

Repeated poles are suported but they must be listed in order.

The output is a list of the values of the fraction in the same order as the poles were entered.

**Why am I doing this?**

The HP48 **PF** program came in clutch when I was in engineering school in the mid-90s, and it saved my hide more than once during homework and tests. This is my way of saying thank you and giving back.

**Main Program:**

**Pre-requisite:**

Matrix A must be allocated as a 2xN array in the following format:

* Row 1 has the **coefficents** of the polynomial in the numerator (but A[1,1] must always be 0.0)[[1]](#footnote-2)
* Row 2 has the **roots** of the polynomial in the denominator (repeated roots must be listed together)

Using the example from PF on the previous page, Matrix A would look like this:

Size: A [2 5]

[0 0 0 1 5]

[4 -2 1 1 1]

Note that the root of (s-r) is r and the root of (s+r) is -r, so mind the signs when transcribing from transfer function into the A matrix.

**Running Program:**

Once Matrix A is populated, run *Program A* via **GSB A** (or just **A** if in User-Mode)

The program will utilize all 5 matrices during runtime but C, D and E are temporary place-holders and can be safely deallocated to save space.

**Results:**

Answer will be placed in Matrix B of size [1 N]. Continuing with the example, Matrix B would look like this:

Size: B [1 5]

[0.055555 0.0185185 -0.074074 -0.11111 -.6666 ]

Numerically, this expansion would be written down as:

0.055555 0.0185185 -0.074074 -0.11111 -0.6666

--------- + ---------- + ---------- + -------- + -------

(s - 4) (s + 2) (s – 1) (s – 1)2 (s – 1)3

Note: Unlike the original PF, this program will report the expansion for the repeated roots in **ascending** order.

**Memory Limitations:**

Given that the polynomial multiplies are done via matrix multiplies (easier implementation via native functions, namely X\*Y) there is a limitation on the number of partial fractions possible. For **default** mode, denominator denominator cannot be larger than order 4. For **15.2** mode, the largest denominator order is **7**. Since switching modes does not destroy the memory, the user can go between default and 15.2 modes to run the program with no penalty.

1. This is because the order of the numerator must always be less than N. If not, perform a polynomial division first. [↑](#footnote-ref-2)