The package provides a parser \texttt{\poldef} of algebraic polynomial expressions. Once defined, a polynomial is usable by its name either as a numerical function in \texttt{\xintexpr/\xin\texttt{teval}}, or for additional polynomial definitions, or as argument to the package macros.

This document illustrates root localization via usage of macros such as \texttt{\PolToSturm} and \texttt{\PolSturm\texttt{mIsolateZeros}} which implement the Sturm theorem:

- Root localization based on Sturm theorem was added at release 0.4 (2018/02/16).
- Ability to find all rational roots was added at release 0.7.2 (2018/12/09).

As of 0.8 (2021/03/29), \texttt{polexpr} is usable with Plain \TeX{} and not only with \LaTeX{}. The examples here use most of the time a syntax which works with both.

Copying-pasting from pdf the example source may lose formatting. Formerly, they were included verbatim in the html documentation. Here they are both rendered verbatim and got executed during the \LaTeX{} run which created this pdf file, with the output shown after the source code.

Regarding how polynomial coefficients are printed on the typeset page by \texttt{\PolTypeset}:

- The default for \texttt{\PolTypesetOne} is to use \texttt{\xintTeXsignedFrac} with \LaTeX{}, \texttt{\xintTeXsigne\texttt{dOver}} with Plain. See the \texttt{xintexpr} documentation for a description of what these macros do. A sensible definition is:
  \begin{verbatim}
  \def\PolTypesetOne#1{\PolDecToString{\xintREZ{#1}}}%
  \end{verbatim}
  It means to use decimal notation, with perhaps a trailing denominator if the argument is a fraction, and will suppress trailing zeros after the decimal mark.

- As these are expandable macros, they are usable to redefine \texttt{\PolToExprCmd} as well:
  \begin{verbatim}
  \def\PolToExprCmd#1{\PolDecToString{\xintREZ[#1]}}%
  \end{verbatim}
  This will customize the output of \texttt{\PolToExpr} (which a priori is destined for writes to external files but may also be used on the typeset page).

With \texttt{xintverbosetrue} in the \TeX{} source extra information relative to the internal data manipulated by the macros will be written to the .log file.

Package macros related to root localization create (user-level) new polynomials, or numeric variables, via a naming scheme using the given \texttt{<sturmname>} as prefix. It is thus advisable to keep this \texttt{<sturmname>} name-space separate from the one used to name polynomial or scalar variables.
Regrettably all examples here use the condemnable \PolToSturm{f}{f} practice which means that internally defined polynomials will use as prefix the original polynomial name. This merge of namespaces may cause overwriting previously defined data and may lead to hard-to-debug problems.

1 A first example

In this example the polynomial is square-free.

\poldef f(x) := x^7 - x^6 - 2x + 1;
\PolToSturm{f}
\PolSturmIsolateZeros{f}
The \PolTypeset{f} polynomial has \PolSturmNbOfIsolatedZeros{f} distinct real roots which are located in the following intervals:
\PolPrintIntervals{f}
The \PolTypeset{x^7 - x^6 - 2x + 1} polynomial has 3 distinct real roots which are located in the following intervals:
-2 < Z_1 < -1
0.4 < Z_2 < 0.5
1 < Z_3 < 2

Here is the second root with ten more decimal digits:
\PolRefineInterval[10]{f}{2}$$\PolSturmIsolatedZeroLeft{f}{2} < Z_2 < \PolSturmIsolatedZeroRight{f}{2}$$
Here is the second root with ten more decimal digits:
0.49623869487 < Z_2 < 0.49623869488

And here is the first root with twenty digits after decimal mark:
\PolEnsureIntervalLength{f}{1}{-20}$$\PolSturmIsolatedZeroLeft{f}{1} < Z_1 < \PolSturmIsolatedZeroRight{f}{1}$$
And here is the first root with twenty digits after decimal mark:
-1.07196788410802660342 < Z_1 < -1.07196788410802660341

The first element of the Sturm chain has degree \PolDegree{f_0}. As this is the original degree \PolDegree{f} we know that \PolDegree{f} is square free. Its derivative is up to a constant \PolTypeset{f_1} (in this example it is identical with it).
\PolToSturm{f_1}{f_1}\PolSturmIsolateZeros{f_1}
The derivative has \PolSturmNbOfIsolatedZeros{f_1} distinct real roots:
\PolPrintIntervals[W]{f_1}
\PolEnsureIntervalLengths{f_1}{-10}
Here they are with ten digits after decimal mark:
\PolPrintIntervals[W]{f_1}
The first element of the Sturm chain has degree 7. As this is the original degree 7 we know that \( f \) is square free. Its derivative is up to a constant \( 7x^6 - 6x^5 - 2 \) (in this example it is identical with it). The derivative has 2 distinct real roots:

\[
-0.8 < W_1 < -0.7 \\
1.06 < W_2 < 1.07
\]

Here they are with ten digits after decimal mark:

\[
-0.7113598308 < W_1 < -0.7113598307 \\
1.0653438848 < W_2 < 1.0653438849
\]

\[
\begin{align*}
\text{PolDiff}\{f_1\}\{f''\} \\
\text{PolToSturm}\{f''\}\{f''\} \\
\text{PolSturmIsolateZeros}\{f''\}
\end{align*}
\]

The second derivative is \text{PolTypeset}\{f''\}. It has \text{PolSturmNbOfIsolatedZeros}\{f''\} distinct real roots:

\[
\text{PolPrintIntervals}\{X\}\{f''\}\%
\]

Here is the positive one with 20 digits after decimal mark:

\[
\text{PolEnsureIntervalLength}\{f''\}\{2\}\{-20\}\% \\
X_2 = \text{PolSturmIsolatedZeroLeft}\{f''\}\{2\}\dots
\]

The second derivative is \( 42x^5 - 30x^4 \). It has 2 distinct real roots:

\[
X_1 = 0 \\
0.7 < X_2 < 0.8
\]

Here is the positive one with 20 digits after decimal mark:

\[
X_2 = 0.71428571428571428571 \ldots
\]

2 A degree four polynomial with nearby roots

Notice that this example is a bit outdated as 0.7 release has added \text{PolSturmIsolateZeros}\{<sturmname>\} which would find exactly the roots. The steps here retain their interest when one is interested in finding isolating intervals for example to prepare some demonstration of dichotomy method.

\[
\text{PolDef}\{Q\}\{(x-1.050001)(x-1.105001)(x-1.110501)(x-1.111051)\} \\
\text{PolTypeset}\{Q\} \\
\text{PolToSturm}\{Q\}\{Q\}\% \\
\text{PolSturmIsolateZeros}\{Q\} \\
\text{PolPrintIntervals}\{Q\}
\]

\[
x^4-4376554.10^{-6}x^3+7181524404656.10^{-12}x^2-5236452059410679654.10^{-18}x+1431546080701127890651551.10^{-24}
\]

\[
1.05 < Z_1 < 1.06 \\
1.10 < Z_2 < 1.11 \\
1.110 < Z_3 < 1.111 \\
1.111 < Z_4 < 1.112
\]

\[
\text{PolRefineInterval}\{Q\}\{1\}
\]
\PolRefineInterval*{Q}{2}
\PolRefineInterval*{Q}{3}
\PolRefineInterval*{Q}{4}
\PolPrintIntervals{Q}
\PolEnsureIntervalLengths{Q}{-6}
\PolPrintIntervals{Q}
% finds here all roots exactly
\begin{align*}
Z_1 &= 1.050001 \\
1.105 &< Z_2 < 1.106 \\
1.1105 &< Z_3 < 1.1106 \\
1.11105 &< Z_4 < 1.11106
\end{align*}

% defines here all roots exactly
\begin{align*}
Z_1 &= 1.050001 \\
Z_2 &= 1.105001 \\
Z_3 &= 1.110501 \\
Z_4 &= 1.111051
\end{align*}

3 The degree nine polynomial with 0.99, 0.999, 0.9999 as triple roots

Define a user command (\texttt{xinttools} is loaded automatically by \texttt{polexpr}):
\begin{verbatim}
def\showmultiplicities#1{% #1 = "sturmname"
xintFor* ##1 in {\xintSeq{1}{\PolSturmNbOfIsolatedZeros{#1}}}}\do{%
The multiplicity is $\PolSturmIsolatedZeroMultiplicity{#1}{##1}$
\PolSturmIfZeroExactlyKnown{#1}{##1}{
[at the root $x=\PolSturmIsolatedZeroLeft{#1}{##1}$]$
\PolSturmIsolatedZeroLeft{#1}{##1}<x<\PolSturmIsolatedZeroRight{#1}{##1}$]
{for the root such that}
$\PolSturmIsolatedZeroLeft{#1}{##1}<x<\PolSturmIsolatedZeroRight{#1}{##1}$}
par
%
%}
\end{verbatim}

\def\PolDef{f}{(x-0.99)^3(x-0.999)^3(x-0.9999)^3}
\PolTypeset{f}
\PolToSturm{f}{f}% it is allowed to use "polname" as "sturmname" too
\PolSturmIsolateZerosAndGetMultiplicities{f} it is allowed to use "polname" as "sturmname" too
\PolSturmIsolatedZerosAndGetMultiplicities\par
\PolSturmIsolateZeros*{f} which is exactly the same, but shorter..

\PolDef{g}{f(x)+1e-27}
\PolTypeset{g}
\PolToSturm{g}{g}
\PolSturmIsolateZeros*{g}

It is fun to modify only a tiny bit the polynomial and see if \texttt{polexpr} survives:
The multiplicity is 1 for the root such that $0.98 < x < 0.99$
The multiplicity is 1 for the root such that $0.9991 < x < 0.9992$
The multiplicity is 1 for the root such that $0.9997 < x < 0.9998$

This means that the multiplicity-3 roots each became a real and a pair of complex ones. Let's see them better:

$Q_{norr}(x) = x^2 + 1$

Here, all real roots are rational. Let's get their decimal expansion too:

\begin{verbatim}
% print decimal expansion of the found roots
\end{verbatim}

\begin{verbatim}
\end{verbatim}
5 A Mignotte type polynomial

\PolDef{P}{x^{10} - (10x-1)^2}\% prints it in expanded form
\PolTypeset{P} % we can use same prefix for Sturm chain
\PolToSturm{P}{P} % this polynomial has \PolSturmNbOfIsolatedZeros{P} distinct real roots:
\PolSturmIsolateZeros{P} % finds 4 real roots
\PolPrintIntervals{P} %

This polynomial has 4 distinct real roots:

-2 < Z_1 < -1
0.09 < Z_2 < 0.10
0.1 < Z_3 < 0.2
1 < Z_4 < 2

Let us refine the second and third intervals to separate the corresponding roots:
\PolRefineInterval*{P}{2}% will refine to 0.0999999 < Z_2 < 0.09999991
\PolRefineInterval*{P}{3}% will refine to 0.100001 < Z_3 < 0.100002
\PolPrintIntervals{P} %

-2 < Z_1 < -1
0.0999990 < Z_2 < 0.0999991
0.100001 < Z_3 < 0.100002
1 < Z_4 < 2

Let us now get to know all roots with 10 digits after decimal mark:
\PolEnsureIntervalLengths{P}{-10}\%
\PolPrintIntervals{P} % now all roots are known 10 decimal digits after mark

-1.8024470510 < Z_1 < -1.8024470509
0.0999999000 < Z_2 < 0.0999999001
0.1000010000 < Z_3 < 0.1000010001
1.7523477264 < Z_4 < 1.7523477265

Finally, we display 20 digits of the second root:
\PolEnsureIntervalLength{P}{2}{-20}\% makes Z_2 known with 20 digits after mark
$\PolSturmIsolatedZeroLeft{P}{2} < Z_2 < \PolSturmIsolatedZeroRight{P}{2}$

0.099999900004999650028 < Z_2 < 0.099999900004999650029

6 The Wilkinson polynomial

%\xintverbosetrue % for the curious...
\poldef f(x) := mul((x - i), i = 1..20);
\def\PolTypesetCmdPrefix#1{\allowbreak\xintiiifSgn{#1}{}{+}{+}}%
\def\PolTypesetOne#1{\xintDecToString{#1}}%
\noindent\PolTypeset{f}
\PolToSturm{f}{f}
\PolSturmIsolateZeros{f}
The first polynomial is handled fast enough, but the modified one $f(x) - 2^{-23} x^{19}$ takes about 20x longer.

Its Sturm chain polynomials have integer coefficients with up to 321 digits, whereas (surprisingly perhaps) those of the Sturm chain polynomials derived from $f$ never have more than 21 digits ...

Once the Sturm chain is computed and the zeros isolated, obtaining their decimal digits is relatively faster. Here are the ten real roots of $f(x) - 2^{-23} x^{19}$ which would be computed by the commented-out code above:

```
Z_1 = 0.9999999999...
Z_2 = 2.0000000000...
Z_3 = 2.9999999999...
Z_4 = 4.0000000002...
Z_5 = 4.9999999275...
```

The first polynomial is handled fast enough, but the modified one $f(x) - 2^{-23} x^{19}$ takes about 20x longer.

Its Sturm chain polynomials have integer coefficients with up to 321 digits, whereas (surprisingly perhaps) those of the Sturm chain polynomials derived from $f$ never have more than 21 digits ...

Once the Sturm chain is computed and the zeros isolated, obtaining their decimal digits is relatively faster. Here are the ten real roots of $f(x) - 2^{-23} x^{19}$ which would be computed by the commented-out code above:

```
Z_1 = 0.9999999999...
Z_2 = 2.0000000000...
Z_3 = 2.9999999999...
Z_4 = 4.0000000002...
Z_5 = 4.9999999275...
```
7 The second Wilkinson polynomial

\poldef f(x) := \text{mul}(x - 2^{-i}, i = 1..20);

%\PolTypeset{f}
\PolToSturm{f}{f}
\PolSturmIsolateZeros**{f}
\PolPrintIntervals{f}

\begin{align*}
Z_1 &= 0.00000095367431640625 \\
Z_2 &= 0.0000019073486328125 \\
Z_3 &= 0.000003814697265625 \\
Z_4 &= 0.00000762939453125 \\
Z_5 &= 0.0000152587890625 \\
Z_6 &= 0.000030517578125 \\
Z_7 &= 0.00006103515625 \\
Z_8 &= 0.0001220703125 \\
Z_9 &= 1/4096 \\
Z_{10} &= 1/2048 \\
Z_{11} &= 1/1024 \\
Z_{12} &= 1/512 \\
Z_{13} &= 1/256 \\
Z_{14} &= 1/128 \\
Z_{15} &= 0.015625 \\
Z_{16} &= 0.03125 \\
Z_{17} &= 0.0625 \\
Z_{18} &= 0.125 \\
Z_{19} &= 0.25 \\
Z_{20} &= 0.5
\end{align*}

This takes more time than the polynomial with 1, 2, ..., 20 as roots but less than the latter modified by the $2^{-23}$ tiny change to one of its coefficient.

There is some incoherence in output format which has its source in the fact that some roots are found in branches which can only find decimal roots, whereas some are found in branches which could find general fractions and they use \texttt{xintIrr} before storage of the found root. This may evolve in future.

8 The degree 41 polynomial with -2, -1.9, -1.8, ..., 0, 0.1, ..., 1.9, 2 as roots

\PolDef{P}{\text{mul}((x-i\times1e-1), i=-20..20)} \quad \text{i/10 is same but less efficient}
In the defining expression we could have used \( i/10 \) but this gives less efficient internal form for the coefficients (the 10's end up in denominators).

\begin{verbatim}
\def\PolToExprCmd#1{\PolDecToString{\xintREZ{#1}}}
\def\PolToExprTermPrefix#1{\newline\xintiiifSgn{#1}{}{+}{+}}
\def\PolToExprTimes{${}\cdot{}$}
\ttfamily
\PolToExpr{P}
\end{verbatim}

\[
x^41 - 28.7 \cdot x^39 + 375.7117 \cdot x^37 - 2975.11006 \cdot x^35 + 15935.28150578 \cdot x^33 - 61167.527674162 \cdot x^31 + 173944.25936417394 \cdot x^29 - 373686.963560544648 \cdot x^27 + 613012.06658846445 \cdot x^25 - 771182.3113138163125495 \cdot x^23 + 743263.86672885754888959569 \cdot x^21 - 545609.0765994828296371978698 \cdot x^{19} + 301748.325708943677229642930528 \cdot x^{17} - 123655.898766945043469869844544 \cdot x^{15} + 36666.1782054884005855608205864192 \cdot x^{13} - 7607.85821367459445649518380016128 \cdot x^{11} + 1053.15135918687298508885950223794176 \cdot x^{9} - 96.6380005918141132650786081964032 \cdot x^{7} + 4.33701563847327366842552218288128 \cdot x^{5} - 0.0944779968420804735498178265088 \cdot x^{3} + 0.000591901218399276854174416896 \cdot x
\]

which shows coefficients with up to 36 significant digits...

Stress test: not a hard challenge to \texttt{xint + polexpr}, but be a bit patient!

\begin{verbatim}
\PolDef{P}{mul((x-i*1e-1), i=-20..20)}
\PolToSturm{P}{S} % dutifully computes \( S_0, \ldots, S_{41} \)
% the [1] optional argument limits the search to interval (-10,10)
\PolSturmIsolateZeros[1]{S} % finds *exactly* (but a bit slowly) all 41 roots!
\PolPrintIntervals{S} % nice, isn't it?
\end{verbatim}

% Unfortunately \PolPrintIntervals uses a non-breakable array environment
% But see next section on how to customize \PolPrintIntervals and let it
% allow pagebreaks
$Z_1 = -2$
$Z_2 = -1.9$
$Z_3 = -1.8$
$Z_4 = -1.7$
$Z_5 = -1.6$
$Z_6 = -1.5$
$Z_7 = -1.4$
$Z_8 = -1.3$
$Z_9 = -1.2$
$Z_{10} = -1.1$
$Z_{11} = -1$
$Z_{12} = -0.9$
$Z_{13} = -0.8$
$Z_{14} = -0.7$
$Z_{15} = -0.6$
$Z_{16} = -0.5$
$Z_{17} = -0.4$
$Z_{18} = -0.3$
$Z_{19} = -0.2$
$Z_{20} = -0.1$
$Z_{21} = 0$
$Z_{22} = 0.1$
$Z_{23} = 0.2$
$Z_{24} = 0.3$
$Z_{25} = 0.4$
$Z_{26} = 0.5$
$Z_{27} = 0.6$
$Z_{28} = 0.7$
$Z_{29} = 0.8$
$Z_{30} = 0.9$
$Z_{31} = 1$
$Z_{32} = 1.1$
$Z_{33} = 1.2$
$Z_{34} = 1.3$
$Z_{35} = 1.4$
$Z_{36} = 1.5$
$Z_{37} = 1.6$
$Z_{38} = 1.7$
$Z_{39} = 1.8$
$Z_{40} = 1.9$
$Z_{41} = 2$

Release 0.5 has experimental addition of optional argument $E$ to \texttt{PolSturmIsolateZeros}. It instructs to search roots only in interval $(-10^E, 10^E)$. Important: the extremities are assumed to not be roots. In this example, the [1] in \texttt{PolSturmIsolateZeros[1]{S}} gives some speed gain; without it, it turns out in this case that \texttt{polexp} would have started with $(-10^6, 10^6)$ interval.

Please note that this feature may be removed or modified.
9 Roots of a Chebyshev polynomial

\[ T_0(x) := 1; \]
\[ T_1(x) := x; \]
\[ \text{count@ 11} \]
\[ \xintloop \]
\[ \poldef T_{\the\count@}(x) := 2xT_{\the\numexpr\count@-1\relax} - T_{\the\numexpr\count@-2\relax}; \]
\[ \ifnum\count@<15 \advance\count@ 1 \repeat \]
\[ \text{count@ 12} \]
\[ T_{15} = \text{PolTypeset}[X]{T_{15}} \]
\[ \text{PolToSturm}{T_{15}} \]
\[ \text{PolSturmIsolateZeros*}{T_{15}} \] "*" as we will want to confirm multiplicity one
\[ \text{PolEnsureIntervalLengths}{T_{15}}{-20} \% ensure 20 decimal digits for each root \]

\[ T_{15} = 16384X^{15} - 61440X^{13} + 92160X^{11} - 70400X^9 + 28800X^7 - 6048X^5 + 560X^3 - 15X \]

Here is now an example of customization. Indeed \PolPrintIntervals default uses array
and thus does not allow page breaks. And it uses \text{Left < Z < Right} as presentation of roots and we would
like here rather \text{Z = decimal expansion}....

\[ \text{0.8.6 adds an internal patch which would allow usage of amsmath environments}\]
\[ \text{like this:}\]
\[ \def\PolPrintIntervalsBeginEnv\{\begin{align*}\}
\[ \def\PolPrintIntervalsEndEnv\{\end{align*}\}
\[ \text{(the problem was that \text{align} evaluates twice its contents so global variables}\]
\[ \text{need a reset at the end of first pass, which is what 0.8.6 took care of)}\]
\[ \%
\[ \text{Let's simply do this:}\]
\[ \def\PolPrintIntervalsBeginEnv\{\begingroup\leftskip3cm\relax\}
\[ \def\PolPrintIntervalsEndEnv\{\par\endgroup\}
\[ \%
\[ \text{The rows are separated by } \text{PolPrintIntervalsRowSeparator which defaults to } \\\n\[ \text{with \text{LaTeX} and } \text{lcr with \text{Plain}. (prior to 0.8.6 it was hardcoded) }\]
\[ \def\PolPrintIntervalsRowSeparator\{\\jot\} \]
\[ \text{And we enter math mode manually at each row, copying pasting from package}\]
\[ \text{defaults with some added mathon/mathoff:}\]
\[ \def\PolPrintIntervalsKnownRoot\{
\[ \Poly\text{PrintIntervalsPrintMultiplicity}\quad
\[ \text{PolPrintIntervalsTheVar}_{\text{PolPrintIntervalsTheIndex}}=\%
\[ \text{PolPrintIntervalsPrintExactZero}
\]
\[ \}
\[ \%
\[ \def\PolPrintIntervalsUnknownRoot\{
\[ \Poly\text{PrintIntervalsPrintMultiplicity}\quad
\[ \text{PolPrintIntervalsTheVar}_{\text{PolPrintIntervalsTheLeftEndPoint}}=\%
\[ {\text{PolPrintIntervalsTheRightEndPoint}}
\]
\PolPrintIntervals{T_{15}}

(mult. 1) $Z_1 = -0.99452189536827333692 \ldots$
(mult. 1) $Z_2 = -0.95105651629515357211 \ldots$
(mult. 1) $Z_3 = -0.86602540378443864676 \ldots$
(mult. 1) $Z_4 = -0.74314482547739423501 \ldots$
(mult. 1) $Z_5 = -0.58778525229247312916 \ldots$
(mult. 1) $Z_6 = -0.40673664307580020775 \ldots$
(mult. 1) $Z_7 = -0.20791169081775933710 \ldots$
(mult. 1) $Z_8 = 0$
(mult. 1) $Z_9 = 0.20791169081775933710 \ldots$
(mult. 1) $Z_{10} = 0.40673664307580020775 \ldots$
(mult. 1) $Z_{11} = 0.58778525229247312916 \ldots$
(mult. 1) $Z_{12} = 0.74314482547739423501 \ldots$
(mult. 1) $Z_{13} = 0.86602540378443864676 \ldots$
(mult. 1) $Z_{14} = 0.95105651629515357211 \ldots$
(mult. 1) $Z_{15} = 0.99452189536827333692 \ldots$