Find the minmal polynomial for $\cos(r\,\pi)$ and $\sin(r\,\pi)$ where r is rational

see http://www.mapleprimes.com/questions/102084-Solutions-By-Radicals-For-Polynomials-

References from that thread:

Carl Devore, http://mathforum.org/kb/message.jspa?messageID=1569138&tstart=0
with a very compact code for conversion (which I tried to understand)

Robert Israel, http://groups.google.com/group/comp.soft-sys.math.maple/browse_thread/thread/66b4a44c1ad95e96
using Chebyshev polynomials for cos and resultant for switching to sinus.

I tried to understand and to find some improvement (though it is not elegant).

Axel Vogt, March 2011

```
> restart: interface(version); Digits:=18;
```

```
                    Classic Worksheet Interface, Maple 12.02, Windows, Dec 10 2008 Build ID 377066
                                              Digits := 18
```

cos

For cos one basically can use the Chebyshev polynomials, having the desired zeros (see for example stuff about integration methods and the needed nodes). Factoring them reduces to find an according factor (done numerically, may be interval arithmetics and 'shake' is more sound - but hairy to use and it would be more easy to increase working precision in case of troubles).

To special things have to obeyed:

In very low degree or special situations Maple spits out rationals or radicals (cos(Pi/3), cos(Pi/4), ...), hence one needs to find the minimal polynomial of an algebraic number given as radical. Astonishingly I did not find usable codes for that and thus use ways provided by Maple. But I do not like PolynomialTools[MinimalPolynomial] (try it for sin(1) to understand why), instead gfun:-algfuntoalgeq seems more convenient.

The other thing is: for rationals beyond +-1 an automatical simplification invents signs to the original expression and this motivated me to write the code for linear expressions of the original task (ok, I also wanted to do some exercise in handling simple functional coding, since else I always omit it).

What I dislike: factorization is used ...

```
> MinPolyCosinus:=proc(C, x::name)
  local r,p,q, sig, somePoly, listOfFactors,  L,j, Poly, COS, linFct,a,b, X;
  #pat:='pat': patmatch(C,s::realcons*cos(fraction::rational*Pi),'pat');pat;

  COS:='COS'; # just to state it more clear
  linFct:=eval(C, cos=' t -> COS');
  linFct:=expand(linFct);

  if type(linFct, polynom(anything, COS)) then
  else
    WARNING("%1 not linear in cos term", linFct);
    return x - C;
  end if;

  if degree(linFct,COS) <= 1 then
  else
    WARNING("%1 not linear in cos term", linFct);
    return x - C;
  end if;

  # C = a*COS + b
  a:= coeff(linFct, COS, 1);
  b:= coeff(linFct, COS, 0);

  if a = 0 then
    #if hastype(b, radical) then WARNING("%1 contains radicals ...", b) end if;
    #return x - C
    if hastype(b, radical) then
      Poly:=gfun:-algfuntoalgeq(b,x(t));
      Poly:= `if`(lcoeff(Poly,x) <> 0, Poly/lcoeff(Poly,x), Poly);
      return sort(Poly);
    else return x - b;
    end if;
  end if;

  if (a=1 and b=0) then # C = 0 + 1*cos
  else
```

```
  #  linFct:= (COS-b)/a;
  #  linFct:= expand(linFct)
  Poly:= MinPolyCosinus( C/a - b/a, X);
  return eval(Poly,X = (x-b)/a)
  end if;

  if type(C, specfunc(satisfies(x-> type(x/Pi, rational)), cos) ) then # go on
  else
    WARNING("%1 not of type cos( rationalNumber * Pi )", C);
    return x - C   # in case it is evaluated to a constant or ...
  end if;

  r:= op(C)/Pi;
  p,q:= numer(r), denom(r);
  sig:=(-1)^p; #type(p,even) ...
  somePoly:=orthopoly[T](q,x) - sig;              #print(`candidate`=somePoly);
  factors(somePoly)[2];
  listOfFactors:=map( uv -> uv[1], %);
  #testFunction:= t -> is( abs(evalf(combine(t))) < Float(10, -Digits) );
  L:=eval(listOfFactors, x=C);
  L:=combine(L); # collaps exponents for cos, cos(x)^k = ...     #print(L);

  for j from 1 to nops(L) do
    if ( abs(evalf(L[j])) < Float(10, -Digits) ) then break end if;
  end do;
  j;
  Poly:=sort(listOfFactors[j]);
  return Poly;                    # highest coeff = 1
  #return Poly/icontent(Poly);  # or integer coefficients
  end proc; #maplemint(%);
```

```
MinPolyCosinus := proc(C, x::name)
local r, p, q, sig, somePoly, listOfFactors, L, j, Poly, COS, linFct, a, b, X;
    COS := 'COS';
    linFct := eval(C, cos = 't → COS');
    linFct := expand(linFct);
    if type(linFct, polynom(anything, COS)) then  else WARNING("%1 not linear in cos term", linFct); return x – C end if;
    if degree(linFct, COS) ≤ 1 then  else WARNING("%1 not linear in cos term", linFct); return x – C end if;
    a := coeff(linFct, COS, 1);
    b := coeff(linFct, COS, 0);
    if a = 0 then
        if hastype(b, radical) then
            Poly := gfun:-algfuntoalgeq(b, x(t)); Poly := `if`(lcoeff(Poly, x) ≠ 0, Poly / lcoeff(Poly, x), Poly); return sort(Poly)
        else return x – b
        end if
    end if;
    if a = 1 and b = 0 then  else Poly := MinPolyCosinus(C / a – b / a, X); return eval(Poly, X = (x – b) / a) end if;
    if type(C, specfunc(satisfies(x → type(x / π, rational)), cos)) then
    else WARNING("%1 not of type cos( rationalNumber * Pi )", C); return x – C
    end if;
    r := op(C) / π;
    p, q := numer(r), denom(r);
    sig := (-1)^p;
    somePoly := orthopoly[T](q, x) – sig;
    factors(somePoly)[2];
    listOfFactors := map(uv → uv[1], %);
    L := eval(listOfFactors, x = C);
    L := combine(L);
    for j to nops(L) do if abs(evalf(L[j])) < Float(10, –Digits) then break end if end do;
    j;
    Poly := sort(listOfFactors[j]);
    return Poly
end proc
```

```
> C := 'cos'(Pi* 8/7); ``=%;
  'MinPolyCosinus(C, x)': '%'=%;
  'eval'(rhs(%), x=C): combine(%): simplify(%) = '%%';
```

$$C := \cos\left(\frac{8\,\pi}{7}\right)$$

$$= -\cos\left(\frac{\pi}{7}\right)$$

$$\mathrm{MinPolyCosinus}(C, x) = -x^3 - \frac{1}{2}x^2 + \frac{1}{2}x + \frac{1}{8}$$

$$0 = \left(-x^3 - \frac{1}{2}x^2 + \frac{1}{2}x + \frac{1}{8}\right)\Big|_{x = -\cos\left(\frac{\pi}{7}\right)}$$

Some brute test: $\cos\left(\frac{\pi\, p}{q}\right)$ for $p = -20 \,..\, 40$, $q = 1 \,..\, 20$:

```
> # Alejandro Jakubi points out, that the 3-argument version for signum is the solid one
  #[-2,0,0.0, 5]; #map(signum, %);
  #map('t -> signum(0,t, 0)', %%);
> # test
  x:='x':
  for q from 1 to 20 do
    for p from -q to 2*q do
      C := cos(Pi* p/q);
      F:=MinPolyCosinus(C, x);
  #print(F);
      eval(F, x=C); combine(%): y:=simplify(%);

      check:= type(F, polynom(rational, x)) and irreduc(F) and signum(0,y,0)=0;
      if not(check) then
        print(C, type(F, polynom(rational, x) ), irreduc(F))
      end if;

    end do;
  end do;
  `done.`;
  y:='y': p:='p': q:='q':
```

done.

```
> 
```

sin

For the sinus there is a way to use the resultant. But that may introduce reducible solutions and one again would have to find and select factors. Instead enforce a conversion to the cosinus, which Maple would not be willing to choose. After that and embedding the core solution in a similar manner as above one can fall back to the solution that we already have.

```
> #sin(x): %=convert(%, cos);
  #x+1/2*Pi: eval(%, x=p/q*Pi): '%'=normal(%);
> sin(p/q*Pi): %=convert(%, cos): subs(cos=COSINUS,'%'): normal(%):
  subs(COSINUS=cos, %);
  expand(%): is(%);
```

$$\sin\left(\frac{\pi\, p}{q}\right) = -\cos\left(\frac{\pi\,(2\,p + q)}{2\,q}\right)$$

true

```
> MinPolySinus:=proc(S, x::name)
  local C, F, G, SIN, linFct,a,b, Poly;

  SIN:='SIN'; # just to state it more clear
  linFct:=eval(S, sin=' t -> SIN');
  linFct:=expand(linFct);

  if type(linFct, polynom(anything, SIN)) then
  else
    WARNING("%1 not linear in sin term", linFct);
    return x - S;
  end if;

  if degree(linFct,SIN) <= 1 then
  else
    WARNING("%1 not linear in sin term", linFct);
    return x - S;
  end if;

  # S = a*SIN + b
  a:= coeff(linFct, SIN, 1);
  b:= coeff(linFct, SIN, 0);

  if a = 0 then
    if hastype(b, radical) then
      Poly:=gfun:-algfuntoalgeq(b,x(t));
      Poly:= `if`(lcoeff(Poly,x) <> 0, Poly/lcoeff(Poly,x), Poly);
      return sort(Poly);
    else return x - b;
    end if;
```

```
  end if;

  convert(S, cos); subs(cos=COSINUS,'%'); normal(%); C:=subs(COSINUS=cos, %); #print(C);

  #C:= subs(sin=cos, S);
  return sort( MinPolyCosinus(C, x) );
  end proc;
MinPolySinus := proc(S, x::name)
local C, F, G, SIN, linFct, a, b, Poly;
    SIN := 'SIN';
    linFct := eval(S, sin = 't → SIN');
    linFct := expand(linFct);
    if type(linFct, polynom(anything, SIN)) then  else WARNING("%1 not linear in sin term", linFct); return x – S end if;
    if degree(linFct, SIN) ≤ 1 then  else WARNING("%1 not linear in sin term", linFct); return x – S end if;
    a := coeff(linFct, SIN, 1);
    b := coeff(linFct, SIN, 0);
    if a = 0 then
        if hastype(b, radical) then
            Poly := gfun:-algfuntoalgeq(b, x(t)); Poly := `if`(lcoeff(Poly, x) ≠ 0, Poly / lcoeff(Poly, x), Poly); return sort(Poly)
        else return x – b
        end if
    end if;
    convert(S, cos);
    subs(cos = COSINUS, '%');
    normal(%);
    C := subs(COSINUS = cos, %);
    return sort(MinPolyCosinus(C, x))
end proc
```

```
> S:= 'sin'(7/10*Pi); ``=%;
  #C:=subs(sin=cos, S);
  #F:=MinPolyCosinus(C,x); irreduc(F);
  G:='MinPolySinus(S,x)'; ``=%;
  'eval'(G, x=S): combine(%): simplify(%) = '%%';
  'irreduc(G)': '%'=%;
```

$$S := \sin\left(\frac{7\,\pi}{10}\right)$$

$$= \sin\left(\frac{3\,\pi}{10}\right)$$

$$G := \mathrm{MinPolySinus}(S, x)$$

$$= x^2 - \frac{1}{2}x - \frac{1}{4}$$

$$0 = \left(x^2 - \frac{1}{2}x - \frac{1}{4}\right)\Bigg|_{x = \sin\left(\frac{3\,\pi}{10}\right)}$$

$$\mathrm{irreduc}(G) = \mathrm{true}$$

Same brute test, now for the sinus

```
> # test
  x:='x':
  for q from 1 to 20 do
    for p from -q to 2*q do
      S := sin(Pi* p/q);
      G:=MinPolySinus(S, x); #print(type(G, polynom(rational, x) ), irreduc(G));
  #print(G);
      eval(G, x=S); combine(%): y:=simplify(%);
      #if signum(y) <> 0 then print(p/q, S) end if;

      check:= type(G, polynom(rational, x)) and irreduc(G) and signum(y)=0;
      if not(check) then
        print(S, type(G, polynom(rational, x) ), irreduc(G))
      end if;

    end do:
  end do:
  `done.`;
  y:='y': p:='p': q:='q':
```

done.

Some more examples:

```
> for q in [26,34,42] do
    S:=  'sin'(Pi* 1/q);
    G:=MinPolySinus( S , x):
    'eval(G, x=S)' =  simplify(combine(eval(G, x=S)));
    print(`irreducible` = irreduc(G));
    ``;
  end do; q:='q':
```

$$S := \sin\left(\frac{\pi}{26}\right)$$

$$G := x^6 + \frac{1}{2}x^5 - \frac{5}{4}x^4 - \frac{1}{2}x^3 + \frac{3}{8}x^2 + \frac{3}{32}x - \frac{1}{64}$$

$$G\Big|_{x=S} = 0$$

irreducible = true

$$S := \sin\left(\frac{\pi}{34}\right)$$

$$G := x^8 + \frac{1}{2}x^7 - \frac{7}{4}x^6 - \frac{3}{4}x^5 + \frac{15}{16}x^4 + \frac{5}{16}x^3 - \frac{5}{32}x^2 - \frac{1}{32}x + \frac{1}{256}$$

$$G\Big|_{x=S} = 0$$

irreducible = true

$$S := \sin\left(\frac{\pi}{42}\right)$$

$$G := x^6 - \frac{1}{2}x^5 - \frac{3}{2}x^4 + \frac{3}{4}x^3 + \frac{1}{2}x^2 - \frac{1}{4}x + \frac{1}{64}$$

$$G\Big|_{x=S} = 0$$

irreducible = true

```
> 
```

strange example

```
> S:=  sin(Pi* 1/192);
  G:=MinPolySinus( S , x);
  'eval(G, x=S)': combine(%): '%%'= simplify(%);
```

$$S := \sin\left(\frac{\pi}{192}\right)$$

$$G := x^{64} - 16x^{62} + 122x^{60} - 590x^{58} + \frac{32509}{16}x^{56} - \frac{42427}{8}x^{54} + \frac{697851}{64}x^{52} - \frac{1159587}{64}x^{50} + \frac{202927725}{8192}x^{48} - \frac{57803655}{2048}x^{46}$$
$$+ \frac{443161355}{16384}x^{44} - \frac{359546005}{16384}x^{42} + \frac{7937669495}{524288}x^{40} - \frac{2334608675}{262144}x^{38} + \frac{9378456563}{2097152}x^{36} - \frac{4019338527}{2097152}x^{34} + \frac{751616304549}{1073741824}x^{32}$$
$$- \frac{29161583781}{134217728}x^{30} + \frac{61281589105}{1073741824}x^{28} - \frac{13546456539}{1073741824}x^{26} + \frac{80047243185}{34359738368}x^{24} - \frac{6116566755}{17179869184}x^{22} + \frac{6116566755}{137438953472}x^{20}$$
$$- \frac{616197075}{137438953472}x^{18} + \frac{6285210165}{17592186044416}x^{16} - \frac{96695541}{4398046511104}x^{14} + \frac{35624673}{35184372088832}x^{12} - \frac{1176791}{35184372088832}x^{10}$$
$$+ \frac{840565}{1125899906842624}x^8 - \frac{5797}{562949953421312}x^6 + \frac{341}{4503599627370496}x^4 - \frac{1}{4503599627370496}x^2 + \frac{1}{18446744073709551616}$$

$$G\Big|_{x=S} = 0$$

```
> RootOf(G,x):
  tmp:=[allvalues(%)]:
  map( 't -> t-S', %):
  evalf(%);
```

$$[-0.88\ 10^{-17}, 0.0653593425071814413, 0.0979252333403596183, 0.162655129650145897, 0.194749820732478382, 0.258226886558445558,$$
$$0.289541288470066683, 0.351154204968216786, 0.381386742900524268, 0.440542144003933893, 0.469401662089853276,$$
$$0.525529848948264936, 0.552738414252411448, 0.605298841743590626, 0.630594420908370583, 0.679080903383124870,$$
$$0.702219886153211273, 0.746165472279901316, 0.766925017602163583, 0.805906487363288328, 0.824086669467951238,$$
$$0.857728610000272070, 0.873154343795369253, 0.901132764821004528, 0.913655492057525333, 0.935700946087437473,$$
$$0.945200066056475166, 0.961100243317085083, 0.967484274300590633, 0.977086047392957613, 0.980293507682693540,$$
$$0.983504406283075003, -0.0327234632529735546, -0.0980828057601550047, -0.130648696593333182, -0.195378592903119461,$$
$$-0.227473283985451946, -0.290950349811419122, -0.322264751723040247, -0.383877668221190350, -0.414110206153497832,$$
$$-0.473265607256907457, -0.502125125342826840, -0.558253312201238500, -0.585461877505385012, -0.638022304996564190,$$
$$-0.663317884161344147, -0.711804366636098434, -0.734943349406184837, -0.778888935532874880, -0.799648480855137147,$$
$$-0.838629950616261892, -0.856810132720924802, -0.890452073253245634, -0.905877807048342817, -0.933856228073978092,$$

-0.946378955310498897, -0.968424409340411037, -0.977923529309448730, -0.993823706570058647, -1.00020773755356420, -1.00980951064593118, -1.01301697093566710, -1.01622786953604857]

```
> S=tmp[1]; evalf[100](%): fnormal(%); is(%);
```

$$\sin\left(\frac{\pi}{192}\right) = \frac{\sqrt{8-2\sqrt{8+2\sqrt{8+2\sqrt{8+2\sqrt{6}+2\sqrt{2}}}}}}{4}$$

$$0.016361731626486781 6 = 0.016361731626486781 6$$

true

Now torture somebody by asking for a de-nesting of that. And be faced with a consequence of pagan's buffling suggestion:

```
> 'S'=convert(S, exp); ``=expand(rhs(%)); #radnormal(%);
  simplify(%, radical);
  evalf[100](%): fnormal(%);
```

$$S = \frac{-1}{2} I \left(\mathbf{e}^{(1/192\, I\, \pi)} - \mathbf{e}^{\left(\frac{-1}{192} I \pi\right)} \right)$$

$$= -\frac{1}{2} I (-1)^{(1/192)} - \frac{1}{2} I (-1)^{\left(\frac{191}{192}\right)}$$

$$= -\frac{(-1)^{\left(\frac{97}{192}\right)}}{2} + \frac{(-1)^{\left(\frac{95}{192}\right)}}{2}$$

$$= 0.016361731626486781 6 + 0.\, I$$

```
> 
```