Janne Koljonen
Vaasa yliopisto
5.4.2012
AUTO1030 Signaalien käsittely
Harjoitus 8, tehtävä 2

Symbolinen polynomien jakolasku

Rationaalipolynomi on: $p(x) = \frac{4x^3 + 3x^2 + 2x + 1}{2x^4 + 3x^3 + 2x^2 + 3x + 2}$. Jos x:n paikalla olisi z, p kuvaisi IIR-suotimen siirtofunktiota. Rationaalipolynomin potenssisarjakehitelmä ja vastaavasti IIR-suotimen impulssivaste saadaan jakokulmalaskulla:

$$2x^4 + 3x^3 + 2x^2 + 3x + 2 \mid \overline{4x^3 + 3x^2 + 2x + 1}$$

Jaetaan $4x^3/2x^4 = 2x^{-1}$:

$$2x^4 + 3x^3 + 2x^2 + 3x + 2 \mid \underline{\qquad\qquad\qquad} \overline{2x^{-1}}$$
$$4x^3 + 3x^2 + 2x + 1$$

Kerrotaan $2x^{-1}\cdot(2x^4 + 3x^3 + 2x^2 + 3x + 2) = 4x^3 + 6x^2 + 4x + 6 + 4x^{-1}$:

$$2x^4 + 3x^3 + 2x^2 + 3x + 2 \mid \underline{\qquad\qquad\qquad} \overline{2x^{-1}}$$
$$4x^3 + 3x^2 + 2x + 1$$
$$\underline{4x^3 + 6x^2 + 4x + 6 + 4x^{-1}}$$

Vähennetään $(4x^3 + 3x^2 + 2x + 1) - (4x^3 + 6x^2 + 4x + 6 + 4x^{-1})$:

$$2x^4 + 3x^3 + 2x^2 + 3x + 2 \mid \underline{\qquad\qquad\qquad} \overline{2x^{-1}}$$
$$4x^3 + 3x^2 + 2x + 1$$
$$\underline{4x^3 + 6x^2 + 4x + 6 + 4x^{-1}}$$
$$-3x^2 - 2x - 5 - 4x^{-1}$$

Jaetaan $-3x^2/2x^4$:

$$2x^4 + 3x^3 + 2x^2 + 3x + 2 \mid \underline{\qquad\qquad\qquad} \overline{2x^{-1} - 1{,}5\,x^{-2}}$$
$$4x^3 + 3x^2 + 2x + 1$$
$$\underline{4x^3 + 6x^2 + 4x + 6 + 4x^{-1}}$$
$$-3x^2 - 2x - 5 - 4x^{-1}$$

Kerrotaan, vähennetään:

$$2x^4 + 3x^3 + 2x^2 + 3x + 2 \mid \underline{\qquad\qquad\qquad} \overline{2x^{-1} - 1{,}5x^{-2}}$$
$$4x^3 + 3x^2 + 2x + 1$$
$$\underline{4x^3 + 6x^2 + 4x + 6 + 4x^{-1}}$$
$$-3x^2 - 2x - 5 - 4x^{-1}$$
$$\underline{-3x^2 - 4{,}5x - 3 - 4{,}5x^{-1} - 3x^{-2}}$$
$$2{,}5x - 2 + 0{,}5x^{-1} + 3x^{-2}$$

Janne Koljonen
Vaasa yliopisto
5.4.2012
AUTO1030 Signaalien käsittely
Harjoitus 8, tehtävä 2

Jaetaan, kerrotaan vähennetään, jne., (kannattaa harjoitella!):

Toteutetaan vastaava Javalla:

```
      /** Janne Koljonen
Vaasan yliopisto
AUTO1030 */

public class Polynomi
{
   // Pääohjelmametodi: testaa
   public static void main(String args[]) throws Exception
   {
      // Kirjoittaa
      System.out.println("Testausta");

      // Luodaan testipolynomi
      double[] p={1, 2, 3, 4};
      double[] q={2, 3, 2, 3, 2};
      // Lasketaan ja tulostetaan osamäärä.
      double[] tulos = jakolasku(p, q, 5);
      print(tulos);
   }

   // Laskee symbolisen jakolaskun polynomeilla
   // Parametrit: p = osoittajapolynomin kertoimet, p[0] = vakiotermi, p[1] = x-termi, p[n] = x^n-
termi.
   // q = nimittäjäpolynomin kertoimet, q[0] = vakiotermi, q[1] = x-termi, q[n] = x^n-termi.
   // maxtermit = kuinka monta termiä jakolaskuun maksimissaan sisällytetään
   // returns: double[] = osamäärän kertoimet, [0] = kertoo suurimman eksponentin,
   // [1] = kertoo suurimman eksponentin kertoimen, [2] = seuraava kerroin, jne.
   public static double[] jakolasku(double[] p, double[] q, int maxtermit)
   {
      double[] osamaara = new double[maxtermit+1];

      // Tehdään taulukot yhtä pitkiksi
      double[] jakojaannos = new double[Math.max(p.length, q.length)];
      double[] jakaja = new double[Math.max(p.length, q.length)];
      double[] tulo = new double[Math.max(p.length, q.length)];

      // Alustetaan jakojäännös p:ksi (suurin eksponentti suurimpaan indeksiin)
      int i_p = p.length-1;
      for (int k=jakojaannos.length-1; k>=0; k--)
      {
         if(i_p >= 0)
         {
            jakojaannos[k] = p[i_p];
         }
         i_p--;
      }
      // ja jakaja q:ksi (indeksointi kuten q:ssa)
      for (int k=0; k<q.length; k++)
      {
         jakaja[k] = q[k];
      }

      // Lasketaan ensin ensimmäisen termin eksponentti.
      osamaara[0] = p.length-q.length; // sieventämättä: (p.length+1)-(q.length+1);


      // Sitten jakokulmalaskua toistaen
      for (int i=1; i<osamaara.length; i++)
      {
         // seuraava osamäärän termi on korkein jakojäännöstermi/korkein nimittäjätermi
         osamaara[i] = jakojaannos[jakojaannos.length-1]/jakaja[q.length-1];

         // Tulo lasketaan kertoen viimeisin osamäärätermi ja jakaja (q) keskenään
         // Tässä on taas hankalat indeksit
         int i_tulo = tulo.length-1;
         for (int i_jakaja=q.length-1; i_jakaja>=0; i_jakaja--)
         {
            tulo[i_tulo] = osamaara[i] * jakaja[i_jakaja];
            i_tulo--;
         }
```

```
         // Vähennyslasku: jakojäännös-tulo
         for (int k=jakojaannos.length-1; k>0; k--)
         {
            jakojaannos[k] = jakojaannos[k-1] - tulo[k-1];
         }
         jakojaannos[0] = 0;

      }
      return osamaara;
   }

   // Tulostaa tulospolynomin, jossa p[0] kertoo korkeimman eksponentin
   private static void print(double[] p)
   {
      System.out.println("Polynomi");
      int maxeksponentti = (int)p[0];
      int temp = 0;
      for (int i=1; i<p.length; i++)
      {
         System.out.print(p[i]+"x^"+(maxeksponentti-temp)+" + ");
         temp++;
      }
      System.out.println();
   }
}
```

Yllä olevaa ohjelmaa voidaan hyödyntää polynomin juurten laskemisessa ja siten myös IIR-suotimen napojen laskemiseen. Navathan ovat IIR-polynomin nimittäjän juuret.

1. Selvitetään yksi nollakohta x_0.
2. Jaetaan nimittäjäpolynomi polynomilla $x - x_0$.
3. Tuloksena saadaan matalamman asteen polynomi, jonka kanssa jatketaan kohdasta 1, kunnes kaikki nollakohdat on selvitetty. (2. asteen polynomiin kannattaa käyttää ratkaisukaavaa.)

```
      // Selvitetään nollakohdat polynomista p(x) = x^3 + 4x^2 + x - 6
      double[] p1 = {-6, 1, 4, 1};
      // Arvaamalla: p(-2) = 0  =>
      double[] x0 = {2, 1}; // x0 = x-2
      print(jakolasku(p1, x0, 6));
```

Tuloksena: $x^2 + 2x - 3$.

```
      double[] p2 = {-3, 2, 1};
      // Arvaamalla: p(1) = 0  =>
      double[] x1 = {-1, 1}; // x1 = x+1
      print(jakolasku(p2, x1, 4));
```

Tuloksena: $x + 3$, josta nähdään, että viimeinen nollakohta $x_0 = -3$.a