Morten Stavik Eggen
Georg Villhelm Seip

Den første formelen kjører med n tid.

Ser hvor mange runder vi kan regne på 1 sek

Nedtellings metode
x^n = x(x^n-1)

x	n	runder per sek	tid per kjøring
1.001	5000	71016	14000 nanosek
1.0001	10000	31962	32000 nanosek
2	10	16069669	62 nanosek

NB: 2 av kjøringene har micro sek som måleenhet

Halveringsmetode
x^n = (x^2)^n/2

x	n	runder per sek	tid per kjøring
1.001	5000	11404943	87 nanosek
1.0001	10000	10577440	95 nanosek
2	10	19374835	51 nanosek

Ut ifra disse metodene ser vi at de er relativt like på små utregninger, men etter antall utregninger vokser så er halveringsmetoden definitivt mye mye raskere, mens nedtellings metodens tid stiger lineært som blir ganske lang tid etterhvert.
Årsaken til at at halveringsmetoden er raskere er at den halverer størrelsen på n hver kjøring mens nedtellings metoden kjører n ganger, når vi halverer hver gang får vi O(log n) kompleksitet og når vi teller ned får vi O(n) kompleksitet.

Math.pow

x	n	runder per sek	tid per kjøring
1.001	5000	15230936	65 nanosek
1.0001	10000	15827468	63 nanosek
2	10	17172229	58 nanosek

Den innebygde pow funksjonen er definitivt raskest og bruker ca samme tid på alle oppgavene. Fikk til og med flere runder på et større antall kjøringer, men det er såpass lite at jeg tipper det bare er tilfeldighet.