Computing Square Root: A Simple Approach

Here is a simple method for finding $\sqrt{5}$. Start with a small enough $x$. At each iteration, we want to increase $x$ by some $h$ (and this $h$ may change at each iteration), with the hope to get nearer and nearer to $\sqrt{5}$, but we want to make sure $x + h$ is not "too large". To determine our $h$, we will decrease it till $(x + h)^2$ is not greater than 5.

Pseudocode:

1. Determine step size $h > 0$ (e.g., $h=1.0$)
2. Pick accuracy threshold, $\varepsilon > 0$ (e.g., $\varepsilon = 0.001$)
3. Guess initial $x$ such that $x^2 < 5$ (e.g., $x=0.0$)
4. While $|(x + h)^2 - 5| \geq \varepsilon$:
   1. while $(x + h)^2 > 5$:
      1. $h \leftarrow h/10$
   2. $x \leftarrow x + h$
Computing Square Root: A Simple Approach

Here is a simple method for finding $\sqrt{5}$. Start with a small enough $x$. At each iteration, we want to increase $x$ by some $h$ (and this $h$ may change at each iteration), with the hope to get nearer and nearer to $\sqrt{5}$, but we want to make sure $x + h$ is not ”too large”. To determine our $h$, we will decrease it till $(x + h)^2$ is not greater than 5.

Pseudocode:

1. Determine step size $h > 0$ (e.g., $h=1.0$)
2. Pick accuracy threshold, $\varepsilon > 0$ (e.g., $\varepsilon = 0.001$)
3. Guess initial $x$ such that $x^2 < 5$ (e.g., $x=0.0$)
4. While $|(x + h)^2 - 5| \geq \varepsilon$:
   1. while $(x + h)^2 > 5$:
      - $h \leftarrow h/10$
   2. $x \leftarrow x + h$
Python’s code:

```python
eps=.001  # accuracy threshold
def our_solver(x0,y,h=1.0,record_iterations=False):
    """ Tries to find x such that x*x is close to y. Params: x0: initial guess; h: initial step size.""
    x=x0
    if (x**2)>y:
        error_message = 'The initial guess, {0}, is too high.'.format(x)
        raise ValueError(error_message)
    estimates_so_far = [x0]
    while abs((x+h)**2-y)>=eps:
        # Want to increase x by some h, but we want to
        # make sure x+h is not "too large".
        # To determine our h, we will
        # decrease it till (x+h)^2 is not greater than y.
        while (x+h)**2>y:
            h /= 10  # decrease h by a factor of 10
        # Now we can increase x.
        x += h  # increase x by h
        estimates_so_far.append(x)
    if record_iterations:
        return x,estimates_so_far
    return x
```

A simple iterative method to approximate $\sqrt{5}$

![Diagram showing iteration points $(x_0, x_1, x_2, x_3, x_4)$ for approximating $\sqrt{5}$]
epsilon= 0.001
number of iterations: 13
our result: 2.235
Python’s sqrt(5) result: 2.2360679775

Want the following to be close to 5.
Squaring our result: 4.995225
Squaring Python’s sqrt(5): 5.0

<table>
<thead>
<tr>
<th>iteration</th>
<th>x</th>
<th>x^2</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>1.0</td>
<td>1.0</td>
</tr>
<tr>
<td>2</td>
<td>2.0</td>
<td>4.0</td>
</tr>
<tr>
<td>3</td>
<td>2.1</td>
<td>4.41</td>
</tr>
<tr>
<td>4</td>
<td>2.2</td>
<td>4.84</td>
</tr>
<tr>
<td>5</td>
<td>2.21</td>
<td>4.8841</td>
</tr>
<tr>
<td>6</td>
<td>2.22</td>
<td>4.9284</td>
</tr>
<tr>
<td>7</td>
<td>2.23</td>
<td>4.9729</td>
</tr>
<tr>
<td>8</td>
<td>2.231</td>
<td>4.977361</td>
</tr>
<tr>
<td>9</td>
<td>2.232</td>
<td>4.981824</td>
</tr>
<tr>
<td>10</td>
<td>2.233</td>
<td>4.986289</td>
</tr>
<tr>
<td>11</td>
<td>2.234</td>
<td>4.990756</td>
</tr>
<tr>
<td>12</td>
<td>2.235</td>
<td>4.995225</td>
</tr>
</tbody>
</table>
Questions about our method:

1. Does it work? *(i.e., does it converge to the correct answer?)* If not, what are the sufficient or necessarily conditions?
2. How fast does it converge? How many iterations? How many operations are needed?
3. How can we find a correct/good initial guess?
4. How sensitive is the method for small changes (or small errors) in the input?