Optimal Decision Rule in Energy-Endurance Trade-off
EE251 Course Project

Jun Zong
School of Information Science and Technology
ShanghaiTech University
Shanghai, China
Email: zongjun@shanghaitech.edu.cn

***Abstract*—Nowadays the smart phones are adapted in multiple working scenarios. Unlike the traditional mobile phones, smart phones are equipped with powerful functions as well as higher power consumption. As a matter of fact, the power control of a smart phone is among the most important features. This paper proposed a situation where you are in a place of nowhere with your smart phone alone (without power bank, you know). Your phone has limited energy, which has to be distributed to both standby and signal detection to increase your probability of survival. The paper also purposes different energy distribution strategy under various situation and determines the optimal rule accordingly. In the last part of the paper, a simulation of the most representative conditions is presented to show how it really works.**

***Keywords*—Detection, Estimation, Mobile, Power**

I. Introduction

Smart phones have invaded into our lives so deeply that one would be thought to be out of date without a smart phone. Also, smart phones are now so powerful that they can fulfill almost all the demands in daily life. However, mobile devices have their own worries. The more powerful one terminal is, the more energy it consumes at every computation it operates. The recent battery problem on iPhones raised public attention on the power control.

It is obvious that operating at neither full power nor least power all the time is an available solution to maximize the power efficiency of a smart phone. The strategy should vary with the specific situation. Imagine that you are in a place of nowhere and your only possession is your smart phone. You have to survive by sending out signals which consumes your phone's energy based on the current signal condition, and so does the standby of your phone. Every single communication of your phone between the base station will increase your probability of survival according to the time interval between the current and the last one as well as the real signal condition. In order to make as many contacts as possible, your phone has to determine a rule on when to communicate with the base station, and the rest part of this paper will go on formulating and deriving the best strategy under different circumstances.

II. Problem Formulation

The key factor in this problem are the time t and the energy each detection consumes E_D. In order to make this problem more intuitive, it can be assumed that the standby consumption is linear to time. To be more specific, we can fix the following parameters,

$$\begin{cases} P_{standby} &= & 1 \\ E_{total} &= & 10000 \end{cases}$$

It is reasonable to assume that the survival probability P_S that every detection increases is related to time t since multiple communications with the base station within very short time interval won't provide any useful information. So the every increased survival probability can be formulated as

$$p_s = (\frac{log(t+1)}{E_D})^2$$

It is also reasonable to put the E_D in the denominator since it is related to the signal intensity. The better the signal condition is, the less it costs to make a contact and the more information it provides. When the battery runs out($E = 0$), the final survival probability is

$$P_S = \sum p_{si}$$

which is the probability you are going to survive after your smart phone is turned off.

Besides, it should be taken into consideration that every contact should be made with best efficiency, that is to say, the biggest probability/consumption ratio. To compute this, the consumption should be derived first as

$$E_C = P_{standby} \cdot t + E_D$$

and the ratio can be derived as

$$r(t) = \frac{(\frac{log(t+1)}{E_D})^2}{P_{standby} \cdot t + E_D}$$

then the problem becomes

$$\max_t r(t), s.t. E_D$$

It is easy to solve the problem with let the derivation of $r(t)$ to t to 0. The numerical result is

$$t^* = e^{1+2ProductLog(0.27(E_D-1))} - 1(1)$$

The $ProductLog(z)$[1] indicates the solution to $z(x) = xe^x$ and can be calculated numerically. With this solution, the optimal can be calculated as long as E_D is determined. However, as a matter of fact, the next E_D is unknown in most situations. The next part will come up with different ways to estimate E_D and how each will affect the result.

III. Propositions on Estimation of E_D

A. *Proposition I: "Markov"*

This proposition is to take the last E_{Di-1} as the next E_{Di} when no additional information is provided. This is especially efficient when the real E_D is monotonically decreasing(e.g. you are walking in the right direction) or increasing. However, this can be extremely unstable when the signal condition is chaos.

B. *Proposition II: "Bayesian"*

This proposition is to take the mean of all former E_D as the next E_D. If signal is averagely good(or bad) everywhere, this position will be the most useful to maximize the profit of every attempt to contact.

IV. Simulation Results

First of all, the $t^*(E_D)$ is like

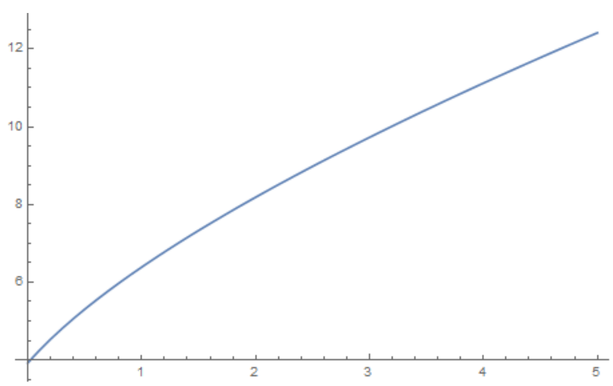

Fig. 1. $t^*(E_D)$ vs. E_D

This shows the relationship between E_D and the optimal t it should make the next contact. In order to test the proposition under different situations, we purposed the following two signal condition along with time.

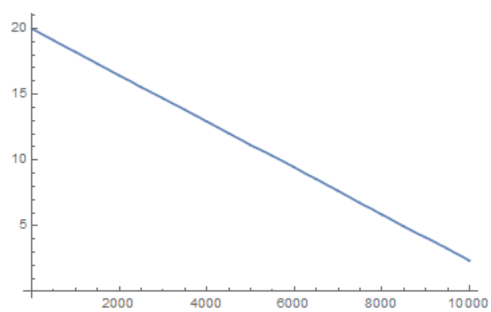

Fig. 2. Monotonically decreasing E_D

The first situation is the monotonically decreasing E_D which simulates that you are going in the right direction where you can get survived.

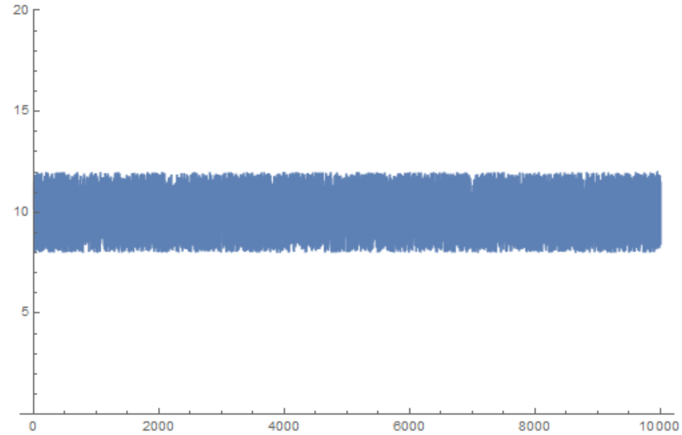

Fig. 3. Random E_D

The second situation is random signal intensity. This simulates a place where every point is averagely disturbed by the noise.

A. *Results of Situation A*

The result of situation A of two propositions separately is as following

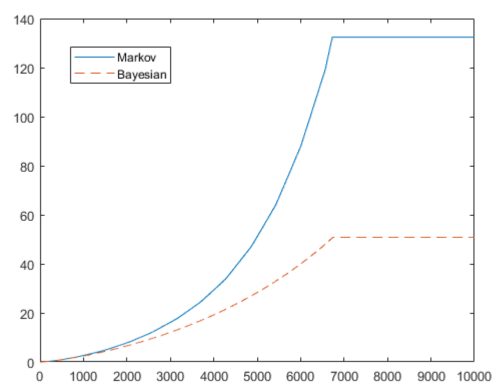

Fig. 4. Result 1

The x-axis denotes the time t and the y-axis indicates the total survival probability P_S. It is clear from the picture that in the first situation, the "Markov" method is way better than the "Bayesian" method, the performance gap between which is almost equal to the worse one. This is mainly due to that the first situation encourages the estimation to step forward to a certain direction, while the mean is lagged because of the property of monotonically decreasing(and also nearly linear).

B. *Results of Situation B*

The result of situation A of two propositions separately is as following

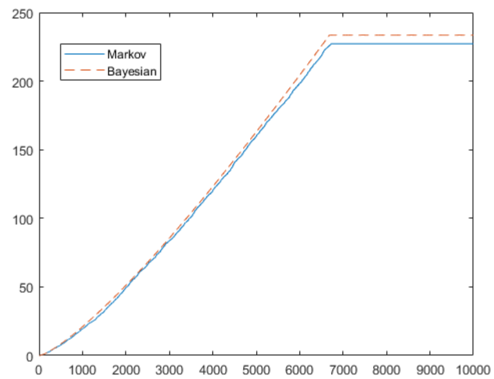

Fig. 5. Result 2

It can be seen from the plot that in situation 2, two methods are very close to each other in performance. This is mainly because the range of E_D is quite small. The two method degenerate to the same when E_D is a constant. Also, with the increase of the range of random E_D, the Bayesian method will perform even better.

V. Conclusion

The current result shows two things: first, the optimal decision rule with given E_D has been derived and can be applied to the real scenario; and second, with different estimators results can be also very different from each other. There is no general optimal estimator for all the situations, and once with the given information, an asymptotic optimal estimator can be derived for the situation. However, calculation and computation also consume energy, so the further work should focus on taking the calculation consumption into consideration.

Appendix A
Proof of (1)

First, take the derivative of the equation $\frac{\frac{ln(t+1)}{E_D}}{t+E_D}$

$$\frac{\partial \frac{\frac{ln(t+1)}{E_D}}{t+E_D}}{\partial t} = \frac{2ln(t+1)}{(t+E_D)E_D^2} \cdot \frac{1}{t+1} - \frac{ln^2(t+1)}{(t+E_D)^2 E_D^2} = 0$$

$$\frac{2}{t+1} = \frac{ln(t+1)}{t+E_D}$$

replace $t+1 = e^y$ into the equation,

$$e^y - 1 + E_D = 0.5ye^y$$

solve it and get

$$y = 1 + 2ProductLog(0.27(E_D - 1))$$

replace y into the original equation,

$$t^* = e^{1+2ProductLog(0.27(E_D-1))} - 1(1)$$

Q.E.D.

Appendix B
The Realization of ProductLog in Matlab

```
function wx=ProductLog(z)
if abs(real(z))+abs(imag(z))<1e-10
  wx=z;
return ;
end
lnz=log(z);
zx=real(lnz);
zy=imag(lnz);

Fy=zeros(2);
F=Fy;
Fx=Fy;
temp=0;
x=zx;
y=zy;
x0=0;
y0=0;
Fy(1)=log(x^2+y^2)/2+x-zx;
Fy(2)=y-zy+atan2(y,x);
erro=abs(Fy(1))+abs(Fy(2));
loopn=1000;
w=1;
while loopn>0 && w>-1.05 && erro>1e-10
    w=x^2+y^2;
    F(1)=x/w+1;
    F(2)=y/w;
    w=F(1)^2+F(2)^2;
    Fx(1)=(Fy(1)*F(1)-Fy(2)*F(2))/w;
    Fx(2)=(Fy(2)*F(1)+Fy(1)*F(2))/w;
    w=1;
    while w>=-1
        x0=x-w*Fx(1);
        y0=y-w*Fx(2);
        Fy(1)=log(x0^2+y0^2)/2+x0-zx;
        Fy(2)=y0-zy+atan2(y0,x0);
        temp=abs(Fy(1))+abs(Fy(2));
        if temp<erro
            erro=temp;
            x=x0;
            y=y0;
            w=-1.03;
        else
            w=w-0.1;
        end
    end
    loopn=loopn-1;
end
wx=x+i*y;
end
```

References

[1] ProductLog,[http://reference.wolfram.com/language/ref/ProductLog.html]