コンピュータ・サイエンス1

第11回
コンピュータでの文字の扱い(2),
アナログ情報とデジタル情報(1)

人間科学科コミュニケーション専攻
白銀 純子

第11回の内容

- コンピュータでの文字の扱い(2)
- アナログ情報とデジタル情報(1)

設問1(問題)

- 1ドル110円とする。この為替レートで、100000円をドルに換えたいとき、下記の2つの問いに答えなさい。

1. 1円あたりのドルの値段を計算し、その上で100000円分のドルの値段を計算しようとすると、1ドル÷110円×100000円という計算式となる。
   この計算方法で、100000円は何ドルに交換できるか答えなさい。
2. 自分がなるべく損をしない(できるだけ多くのドルと交換する)ようにするには、1. の計算式をどのように変更すれば良いか答えなさい。

条件
- 有効桁数が小数点以下第3位までのコンピュータで計算することとする
- 1÷110=0.009090909090909... である

設問1(解答)

- 1. の問題: 1ドル÷110円×100000円
  - 「1÷110」の時点で桁落ちし、0.009
  - 「0.009×100000」の計算をして900
- 2. の問題: 1. の計算式を変更
  - 割り算を後にすると「1×100000÷110」
    - 「100000÷110」で桁落ちして909

解答
1. 900ドル
2. 1×100000÷110

前回の質問の回答

英語のページの文字化け

- 文字化けの原因: ある文字を表す2進数の番号が、規格や環境によって異なるため
  - Ex.
    - ひらがなの「あ」をある規格Aでは$(100)_{10}$、全角の「A」を$(200)_{10}$と表現
    - ひらがなの「あ」をある規格Bでは$(200)_{10}$、全角の「A」を$(100)_{10}$と表現

何らかの理由で、規格Aの番号を使って保存されている文書を規格Bとして開こうとしたとき、文字化け
- ひらがなの「あ」を全角の「A」と表示

- 多バイト文字: 番号の規格を統一できなかった → 文字化けすることがある
- 半角英数: 番号の規格をASCIIに統一できた → 文字化けしない
  - ✓ 1バイト文字でも、ASCIIが使っていない番号を独自に使っている文字がある場合、番号を統一できていなければ文字化けの可能性あり

文字のバイト数

- 半角英数: 1バイト
- 日本語文字: 2バイト
  - ただし、日本語独自の文字コードでは
- 他の国の言葉
  - 言語独自の文字コードの場合は知らないが…世界中の文字をとりまとめて扱う文字コードでは、多くのバイト数を利用
    - この後で具体的に説明

桁数あわせ

- コンピュータでは、常に桁数が重要!
  - ただし、10進数で桁数あわせを求められることはあまりなし
  - 2進数では常に桁数を気にすべし
    - 2桁で表せる2進数を3桁で表す場合は、最も大きな桁に「0」を入れる

期末試験と補講

- 期末試験
  - 用語:「すべて持ち込み不可」なので覚えましょう
- 補講: なし

Question!

前回の復習

ASCII文字(p. 13)

- ASCII: American Standard Code for Information Interchange
- 半角文字を表す文字集合
  - アルファベット大文字(26文字)
  - アルファベット小文字(26文字)
  - 数字(10文字)
  - 記号(スペース,「,」,「.」, etc.)

1文字を表すために、最低限7ビット必要
(6ビット: 64種類の情報, 7ビット: 128種類の情報)

※1文字を表す2進数の桁数(ビット数)は、どの文字でも同じ(つまり7ビット)

ビット数[1](p. 14)

- コンピュータでは8ビットを1つの単位として扱うことが多い
  → ASCII文字も8ビットで表現すると扱いやすい
  - 8ビットのうち、7ビット分(2進数で7桁目まで)で文字を表現する
  - 残りビット(2進数で8桁目)に常に0を入れておく
    - ASCII文字としては無駄なビット
    - 日本語を表現するときに利用

例えば...
A: 65番(10進数)
= 1000001番(2進数)
= 01000001番(2進数, コンピュータ内での表現)
ASCII的には無駄な(何も利用していない・1にはならない)ビット

日本語の文字(p. 15)

- ASCII
  - 1文字を8ビットで表現→全部で256文字分表現可能
  - 現状で128文字存在(128文字分利用されているので残りは128文字)
- 日本語
  - ひらがな: あ～ん(ゐ, ゑなどの旧字を含む), 濁音・半濁音,
    小文字(「ぁ」, 「ぃ」など)
  - カタカナ: ア～ン(ヰ, ヱなどの旧字を含む), 濁音・半濁音(ヴを含む),
    小文字(「ァ」, 「ィ」, 「ヵ」, 「ヶ」など)

169文字

ひらがな・カタカナだけでもASCIIでは表現できない

日本語文字集合の規格(p. 16)

- 現状での日本語文字集合の規格: JIS X 0208:1997
  - ひらがな・カタカナ・漢字・非漢字文字で6879個

JIS第1水準(使用頻度の高い漢字): 2965個
JIS第2水準(使用頻度の低い漢字): 3390個

- 2^{13} = 8192なので、13ビットで表現可能
- コンピュータ処理では、バイト単位(8ビット単位)が好都合

16ビット(2バイト)で日本語1文字を表現

ASCII文字との区別(p. 16)

- 日本語の文書
  - 日本語の2バイト文字
  - ASCIIの1バイト文字

混在

日本語の2バイト文字(JIS X 0208)とASCIIの1バイト文字は区別する必要
(1つの文書の中で、どれが2バイト文字でどれが1バイト文字か)

- モード切り替えによる区別方法
- ASCII文字の番号を避ける区別方法

モード切り替え(p. 16)

- 文字集合切り替えのための特別な記号を用意
  - ここから先はASCII文字
  - ここから先は日本語文字
  - ここから先は中国語漢字
  - etc.

エスケープシーケンス

通常の文書では頻繁に文字集合が切り替わることがなく、同じ文字集合に属する文字が現れることが多いという性質を利用

- 国際標準規格: ISO-2022
- 日本語に適用したもの: ISO-2022-JP

ISO-2022-JPの例(p. 16)

ESC $B	FI	K¥	$N	ESC (B	JP	ESC $B	$9	!#	ESC (B	¥n
	日	本	は		JP		だ	。		

- 「ESC $B」や「ESC (B」、「¥n」などがエスケープシーケンス
- 「FI」や「K¥」、「$N」などは、2バイト文字をASCII文字で表現した場合の文字
  (2バイト文字は、1バイト文字2文字の組み合わせで表現できる)

モード切り替えの問題(p. 17)

- 文書を先頭から順番に見ていく場合には問題ない
- 文書を途中から見ていくときに問題が生じる
  - 見始めた途中の文字が、ASCII文字か日本語文字か、エスケープシーケンスかが判別できない

Ex. 見始めた途中の文字が「70」番だった場合
- ASCII文字の「F」?
- 日本語文字の一部?
- 韓国語の一部?

↓

検索や置換などの文書処理に時間がかかる

ASCII文字の番号を避ける(p. 17)

- ASCIIで使われていない番号を2バイト文字の番号にあてる方法
  - EUC(日本語のものをEUC-JP)
    - 第1バイト(前半の8ビット)と第2バイト(後半の8ビット)両方でASCII領域が避けられている
  - SJIS(Shift JIS)
    - 第2バイト(後半の8ビット)ではASCII領域も使われている
  - 文章のバイト数は、単純に、日本語文字で2バイト、ASCII文字で1バイトで数えれば良い

EUCとSJIS[1](p. 17)

- ASCII文字: 8個の0と1で、1文字分を表現
  - 実際には、7個の0と1で1文字分を表現
  - 8ビット目は必ず0

必ず0

0000000からmmmmまで、フルに使ってASCII文字を1文字ずつ表現

XXXXXXXX

= ASCII文字は、0XXXXXXX という形
Ex.「a」を0と1で表現すると: 01100001

↓

8ビット目が1になる番号(1XXXXXXXという形の番号)はASCII文字ではない

EUCとSJIS[2](p. 17)

- ASCIIで使われていない番号を2バイト文字の番号にあてる方法
  - EUC(日本語のものをEUC-JP)
    - 第1バイト(前半の8ビット)と第2バイト(後半の8ビット)両方でASCII領域が避けられている
  - SJIS(Shift JIS)
    - 第2バイト(後半の8ビット)ではASCII領域も使われている
  - 文章のバイト数は、単純に、日本語文字で2バイト、ASCII文字で1バイトで数えれば良い

例えばある文章が...
00110110010010100100101011010101
1から始まっているから日本語文字
0から始まっているからASCII文字

Webやメールでの文字化け(p. 18)

- Webページや電子メール
  - 文字コードの指示が文書中に書かれている場合
    「charset=iso-2022-jp」や「charset=Shift_JIS」など

    ↓ ソフトウェアは指示通りに文字コードを解釈し、表示
  - 文字コードの指示が文書中に書かれていない場合

    ↓ ソフトウェアは文書のデータの特徴から文字コードを判別

    ↓

文字化けが発生する場合
- 文書中の文字コードの指示が間違っている場合
- 文字コードの指示がなく、文字コードを判別できなかった場合

やってみよう!

- プリントの文字コード(この授業での演習用文字コード)によると、下記1.～3.は、どの文字コードでどんな言葉を表しているか?
  1. 62B7 2654 4C25 1D7F 720B
  2. 6AD0 FBB8 1D7F DCB5 B2BD 09AE
  3. A88D 1D7F 4C25 EC4C B2BD EC4C 01C1

Unicode

言語圏ごとの文字コード(p. 18)

- これまでの多バイト文字の扱い: 異なる言語圏ごとに文字集合を作成
  - 様々な文字集合ができてしまって不便
    - コンピュータネットワークの国際化が進んだ
    - コンピュータの資源が豊富になった
  - 国際文字集合規格として各文字集合を統一化
    - ASCII, ラテン文字, 日本語, 韓国語, 中国語, ベトナム語, ギリシャ文字, 記号, etc.

Unicode: どの文字を扱うかと文字の符号化の方法を決めた規格
- UCS(Universal multi-octet coded Character Set)でどの文字を扱うかを規定
- UTF(UCS Transformation Format)で文字の符号化の方法を規定

UTF-8(p. 18)

- Unicodeでの代表的な符号化方式(符号化方式はいくつか存在)
- 1文字を1～6バイトの可変長(文字によってバイト数が異なる)で符号化する方式
  - ASCIIやISO-2022-JP、Shift JIS、EUC-JPは1文字を同じバイト数で表現
- OS(WindowsやMacなどのオペレーティングシステム)でファイル名などの内部処理に利用
  - 半角英数を符号化した結果が、ASCII文字と全く同じになるため、従来のシステムと相性が良い

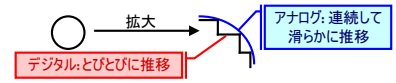
現在、UTF-8への移行が急速に進んでいる
- ただし、以前からのファイルを移行するのは大変なので、完全移行には時間がかかる
- 完全移行できたら、文字化けが起こらなくなる

Question!

アナログ情報とデジタル情報

アナログ情報とデジタル情報(p. 19)

- アナログ情報: 連続的な数値で表現できる情報
  - 物事を表現する数値の桁数が無限
  - Ex. アナログ時計: 1秒から2秒になるまで秒針が止まらず動き続ける(1秒と2秒の間も1.xxxx....秒が存在する)
- デジタル情報: 離散的な数値(とびとびの数値)で表現される情報
  - 物事を表現する数値の桁数が有限
  - Ex. デジタル時計: 1秒の次は2秒(1秒と2秒の間がない)



アナログ/デジタル情報の例(p. 19)

- アナログ情報
  - 現実世界の音
  - カセットテープやレコードに記録された音情報
  - 人の手で描いた絵
  - フィルムに記録された映像
- デジタル情報
  - CDやMDに記録された音情報
  - デジタルカメラが記録した画像情報
  - DVDに記録された音と動画の情報

アナログ情報のデジタル化(p. 20)

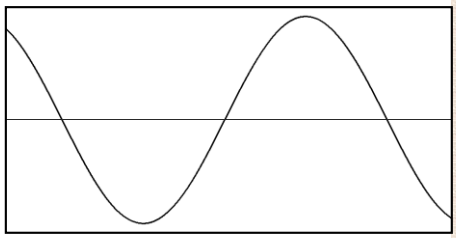
- アナログ情報: 数値化すると連続的
  - 音の強弱の変化
  - 画像の色の濃さ光の強弱の変化

グラフで表すと、なめらかな曲線(正弦波) = アナログ信号

デジタル化するには...
- 標本化
- 量子化

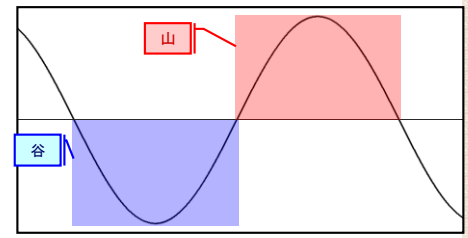
正弦波

- 波の形になっているグラフ
  - いくつかの特徴あり



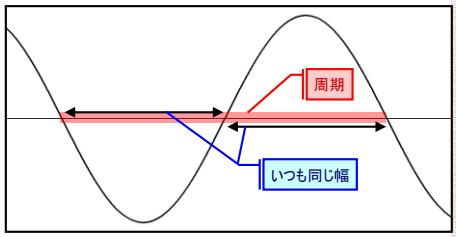
正弦波[特徴その1]

- 山と谷が交互に出現



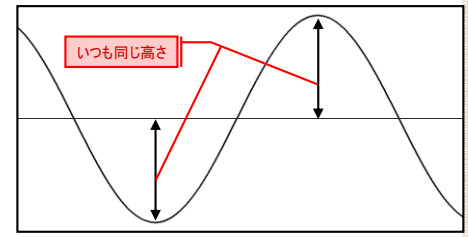
正弦波[特徴その2]

- 山と谷の幅をあわせたものを「周期」と呼び、山と谷の幅はいつも同じ



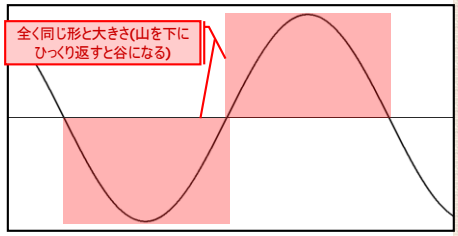
正弦波[特徴その3]

- 山と谷の高さ・低さはいつも同じ



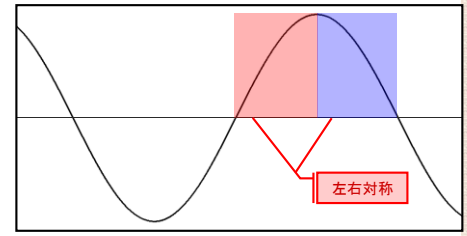
正弦波[特徴その4]

- 山と谷は全く同じ形・同じ大きさ



正弦波[特徴その5]

- 山(谷)は左右対称
  - 山(谷)を真ん中で折りたたむと、きれいに重なる



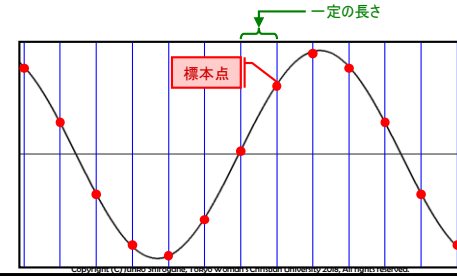
標本化[1](p. 20)

- アナログ信号: 様々な幅や高さの波(正弦波)の組み合わせ
  - 幅の狭い波・広い波、高い波・低い波
- 標本化: アナログ信号の横軸の一定の長さごとに、縦軸の値を調べること
  - 調べた点を「標本点」と呼ぶ
- 調べた以外のところの値は使用しない
  - 標本点の間隔を十分に細かくとれば、もとのアナログ信号の情報を完全に保持できる

どのくらいの間隔で標本化をすれば良いか???

標本化[2](p. 20)

- 標本化: 横軸を一定の長さで区切って縦軸の値を調べること



標本化[3](p. 20)

- 標本化の感覚をどうやって決めるか?
  - 1つの波の幅の半分より小さい間隔で標本点をとれば、元の波形を正確に表現可能(=方程式を算出可能)
  - 1つの波の幅の半分より大きい間隔で標本点をとれば、元の波形とは異なった波形が出現
    エイリアシング

音や画像の情報: 波形の周期の長いもの・短いものを重ね合わせることで表現

波の中で最も狭い幅のものの半分よりも狭い間隔で標本化することが重要

標本化定理[1](p. 21)

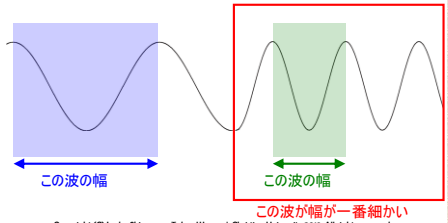
- 標本化定理
  - 最も狭い波の幅の半分よりも狭い幅で標本化すれば、標本から元の波を再現可能

標本化をするときに、どのくらいの間隔で標本点を取れば良いか?を考えるときの目安になる定理

- 間隔が広すぎると、元の波が再現できず、正確に情報をコンピュータに取り込めない
- 間隔が狭すぎると、元の波は再現できるが、情報量が多すぎて、ファイルサイズなどが大きくなる

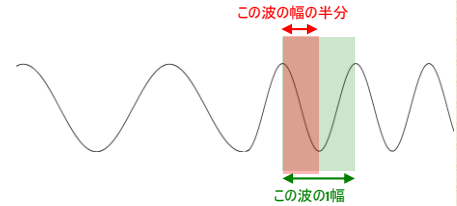
標本化定理[5](p. 21)

- アナログ信号の中のどの波が、一番幅が狭いか?



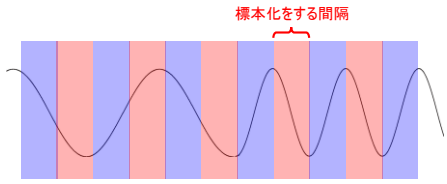
標本化定理[6](p. 21)

- 標本化をする間隔を決定
  1. アナログ信号の中で、最も幅の細かい波を見つける
  2. 1. の波の幅の半分より狭い間隔で標本化をする



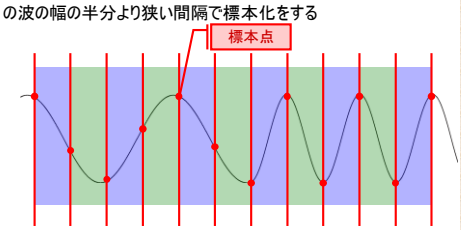
標本化定理[7](p. 21)

- 標本化をする間隔を決定
  1. アナログ信号の中で、最も幅の細かい波を見つける
  2. 1. の波の幅の半分より狭い間隔で標本化をする



標本化定理[8](p. 21)

- 標本化をする間隔を決定
  1. アナログ信号の中で、最も幅の細かい波を見つける
  2. 1. の波の幅の半分より狭い間隔で標本化をする



量子化[1](p. 22)

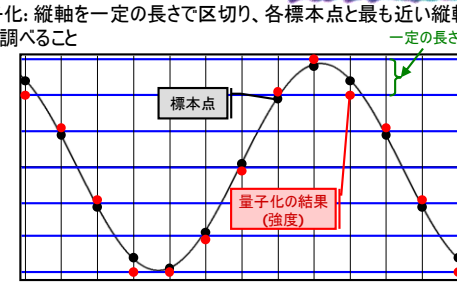
- 量子化: 縦軸の一定の長さごとに、横軸の値を調べること
  1. 横軸の値を調べるための縦軸の量(量子化レベル)をどの程度にするかを決める
  2. 標本化で取り出された標本点のy軸の値(強度)を最も近い量子レベルに置き換える

アナログ情報のデジタル化が完了

- 最終的に、量子化の結果(強度)の値をコンピュータに取り込む
  - もとのアナログの値とは異なる値が取り込まれる

量子化[2](p. 22)

- 量子化: 縦軸を一定の長さで区切り、各標本点と最も近い縦軸の値を調べること



音の符号化[原則](p. 23)

- 音: 空気などの分子の振動現象
- 波の横軸: 音の高さ
  - 人間の耳は、振動の中で20Hz～20kHzのものを聞き分け可能
  - Hz(ヘルツ): 1秒間の振動の回数, または1秒間の標本化の回数
    - 波の幅: 振動の間隔(1秒間の振動の回数)
  - 振動の回数が大きければ(波の幅が小さければ)高い音
  - 振動の回数が小さければ(波の幅が大きければ)低い音
- 波の縦軸: 音の強さ(大きさ)

人間の感覚で感知できる程度の波を扱えれば良い
(人間が聞き分けられない波は扱わなくても良い)

音の符号化[標本化](p. 23)

- 波の横軸: 音の高さ
- 20kHzの波形を再現するには、40kHzで標本化(1秒間に40000回標本化)
  - 20kHzの波: 正弦波の周期は1/20000
    - 標本化は、1/2周期で1回行う
      = 20kHzの波は1/40000間隔で標本化
      = 40kHzで標本化
  - 音楽では高い音も再現する必要: 音楽CDは44.1kHzで標本化
  - 固定電話では人間の声の高さ程度を再現: 8kHzで標本化

音の符号化[量子化](p. 23)

- 波の縦軸: 音の強さ(大きさ)
  - 固定電話: 8ビット(256種類)の量子化レベル
    - 0～255の256段階の数値で音の強さを表現
  - 音楽CD: 16ビット(65,536種類)の量子化レベル
    - 0～65,535段階の数値で音の強さを表現(音の強さをより細かく表現可能)

画像の符号化[1](p. 23)

- 標本化した画像: 点の集まりと考えられる
  - 点: 細かい正方形のマス目

画素(ピクセル, pixel)

  - 画像の長方形のキャンバスは点の集まり
  - 1つ1つの点の大きさによって画像の質が決定
    - 点が大きければ粗い画像
    - 点が小さければきめの細かい画像

標本化の間隔により、点の大きさが決定

  - 1つ1つの点が何色かを記録しておくことで画像を表現

量子化の間隔により、画像中で利用可能な色の種類が決定
(どの程度、微妙な色合いを表現するか)

画像の符号化[2](p. 23)

- カラー画像
  - コンピュータのディスプレイ: 赤(Red), 緑(Green), 青(Blue)の3つの光を利用
    - 3つの光にそれぞれ256段階の濃淡をつけ、3つの光を混ぜ合わせて色を作成
    - 256段階 = 8ビットで表現可能
    - 1つの色: 8ビット × 3つの光 = 24ビットで表現

画像中の1つの点を24ビットで表現

カラー画像: 1つの画素を0～16,777,215までの数値で表現可能

Question!

次回(7月10日)

- 実習をするので24102教室で授業
  - スキャナで取り込んだ写真の標本化・量子化のレベルの違いの確認を実習
- 次回までに写真を1枚コンピュータに取り込んでおくこと
  - やり方はプリント&授業の資料のページで