BIOINFORMATICS

Vol. 00 no. 00 2014
Pages 1–8

Computational Framework for Next-Generation Sequencing of Heterogeneous Viral Populations using Combinatorial Pooling

Pavel Skums [1,*], Alexander Artyomenko [2], Olga Glebova [2], Sumathi Ramachandran [1], Ion Mandoiu [3], David S. Campo [1], Zoya Dimitrova [1], Alex Zelikovsky [2], and Yury Khudyakov [1*]

[1]Division of Viral Hepatitis, Centers of Disease Control and Prevention, Atlanta, Georgia, USA, [2]Department of Computer Science, Georgia State University, Atlanta, Georgia, USA, [3]Department of Computer Science and Engineering, University of Connecticut, Storrs, Connecticut, USA

Received on XXXXX; revised on XXXXX; accepted on XXXXX

Associate Editor: XXXXXXX

ABSTRACT

Motivation: Next-generation sequencing (NGS) allows for analyzing a large number of viral sequences from infected patients, providing an opportunity to implement large-scale molecular surveillance of viral diseases. However, despite improvements in technology, traditional protocols for NGS of large numbers of samples are still highly cost- and labor-intensive. One of the possible cost-effective alternatives is combinatorial pooling. Although a number of pooling strategies for consensus sequencing of DNA samples and detection of SNPs have been proposed, these strategies cannot be applied to sequencing of highly heterogeneous viral populations.

Results: We developed a cost-effective and reliable protocol for sequencing of viral samples, that combines NGS using barcoding and combinatorial pooling and a computational framework including algorithms for optimal virus-specific pools design and deconvolution of individual samples from sequenced pools. Evaluation of the framework on experimental and simulated data for hepatitis C virus showed that it substantially reduces the sequencing costs and allows deconvolution of viral populations with a high accuracy.

Availability: The source code and experimental data sets are available at http://alan.cs.gsu.edu/NGS/?q=content/pooling

Contact: kki8@cdc.gov

1 INTRODUCTION

Next-generation sequencing (NGS) generates a large number of viral sequences carried in samples of infected individuals, offering novel prospects for studying microbial populations and understanding pathogen evolution and epidemiology. NGS provides an opportunity to implement a large-scale molecular surveillance of infectious diseases for monitoring of disease dynamics and providing for informed guidance for planning public health interventions. Although NGS offers a significant increase in throughput, sequencing of viral populations from a large number of specimens is prohibitively expensive and time consuming. Therefore development of a strategy for rapid and cost-effective massive viral sequencing is key to effective molecular surveillance.

Owing to a high mutation rate, RNA viruses exist in infected hosts as highly heterogeneous populations of genetic variants, which are commonly referred to as *quasispecies*. Genetic viral variants can be detected using highly variable subgenomic regions that can be easily amplified and sequenced. Although genetic information presented in short subgenomic regions does not allow for identification of all variants, it is sufficient for transmission networks inference (Holodniy *et al.* (2012); Wertheim *et al.* (2014)), study of drug-resistantce (Campo *et al.* (2014a); Wang *et al.* (2014); Dierynck *et al.* (2014)) and intra-host viral evolution (Ramachandran *et al.* (2011); Palmer *et al.* (2012); Culasso *et al.* (2014)).

To reduce the cost of sequencing of multiple viral samples, multiplexing through barcoding is usually used. Although this is a straightforward approach to a simultaneous sequencing of many viral strains, it requires individual handling of each sample starting from nucleic acid extraction to PCR and library preparation, which increases the costs of sequencing per specimen (Lonardi *et al.* (2013); Duma *et al.* (2013)). Decoding of samples sequenced using large libraries of barcodes may be highly affected by NGS errors (Deakin *et al.* (2014)). Additionally, besides introduction of bias in amplification of different viral variants using PCR primers with different barcodes (that may affect distribution of reads) (Duma *et al.* (2013); Alon *et al.* (2011)), maintaining a large library of barcodes is daunting (Lonardi *et al.* (2013); Duma *et al.* (2013)).

An alternative strategy is combinatorial pooling. Commonly, it was used for tests producing binary results (Du *et al.* (2006); Wu *et al.* (2006); Berman *et al.* (2004)). Recently, several pooling strategies were proposed for more complex assays based on DNA sequencing, SNP calling and a rare alleles detection (Prabhu *et al.* (2009); He *et al.* (2011); Erlich *et al.* (2009); Shental *et al.* (2010); Golan *et al.* (2012); Bansal (2010); Lonardi *et al.* (2013)).

The pooling problem for viral quasipecies sequencing is fundamentally different from all existing pooling protocols. Previously developed methods assume that a single (consensus) sequence must be reconstructed for each sample. In contrast, for viral quasispecies sequencing it is imperative to reconstruct the

*to whom correspondence should be addressed

© Oxford University Press 2014.

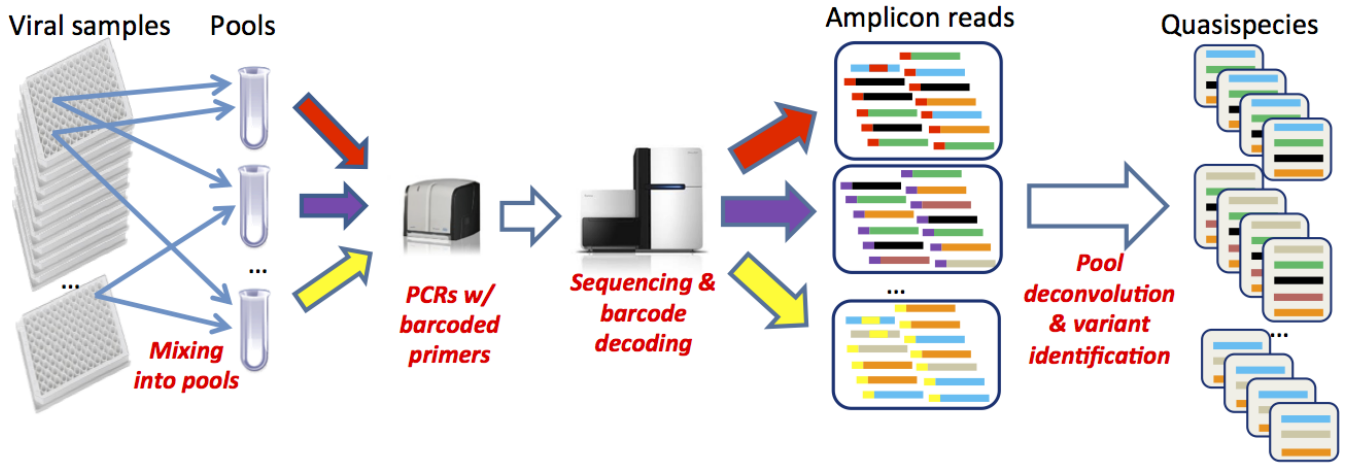

Fig. 1. Combinatorial pooling strategy for viral samples sequencing.

whole population structure of each sample that includes multiple sequence variants and their frequencies. This problem formulation and the nature of heterogeneous viral populations require a completely novel approach for pool design and deconvolution.

We propose a protocol for a cost-effective NGS of complex viral populations, which combines barcoding and pooling and includes the following steps (Fig. 1): (i) mixing samples in a specially designed set of pools so that the identity of each sample is encoded in the composition of pools; (ii) sequencing pools using barcoding; (iii) pools deconvolution; i.e., assignment of viral variants from the pools to individual samples. This approach significantly minimizes the number of PCR and NGS runs, reducing the cost of testing and hands-on time. Additionally, pooling provides opportunity for PCR amplification of viral variants from each sample in different mixtures of samples generated in each pool, thus introducing variation in amplification biases and contributing to sequencing of a more representative set of viral variants from each sample.

Pooling-based sequencing of highly mutable viruses such as human immunodeficiency virus (HIV) and hepatitis C virus (HCV) is particularly difficult because of the complexity of intra-host viral populations, the assessment of which can be distorted by PCR or sampling biases. It is essential to detect both major and minor viral subpopulations, since the latter often have important clinical implications (Skums *et al.* (2012b); Metzner *et al.* (2009); Campo *et al.* (2014a)). Mixing of a large number of specimens or specimens with significant differences in viral titers may contribute to underrepresentation of viral variants from some samples in pools, suggesting that size and composition of pools should be carefully designed. Stochastic sampling from genetically diverse intra-host viral populations usually produces variability in compositions of sets of variants in different pools obtained from the same sample. Additionally, mixing specimens may differentially bias PCR amplification, contributing to mismatching between viral variants sampled from the same host in two pools with different specimen compositions. Thus, straightforward approaches cannot be used for samples deconvolution, indicating that a more complex approach based on clustering techniques is needed. To increase the effectiveness of cluster-based deconvolution and minimize possible clustering errors, it is important to minimize mixing of genetically close samples as can be expected in epidemiologically related samples and samples collected from a small geographic region.

In this paper, we present the first computational framework for combinatorial pool-based sequencing of highly heterogeneous viral samples. The framework includes pools design and pools deconvolution stages. We formulate the pool design problem as an optimization problem and propose a heuristic algorithm to solve it. We propose a method for inference of samples from sequenced pools based on a novel maximum likelihood clustering algorithm for heterogeneous viral samples. We report the results of the framework evaluation using simulated and experimental HCV data.

2 METHODS

2.1 Pools design

The basic idea of the overlapping pools strategy for sequencing n samples is to generate m pools (mixtures of samples) with $m \ll n$ in such a way that every sample is uniquely identified by the pools to which it belongs (Prabhu *et al.* (2009)). Then, after sequencing of pools the obtained amplicon reads can be assigned to samples by a sequence of set-theoretic intersections and differences of pools. The unique assignment is only possible if for any two samples there is a pool *separating* them, i.e., containing exactly one of those samples.

Example. Consider 3 samples S_1, S_2, S_3 and 2 pools $P_1 = S_1 \cup S_2$, $P_2 = S_2 \cup S_3$ (Fig. 2). These pools satisfy the separation requirement, and, therefore, each sample can be recovered, e.g., $S_2 = P_1 \cap P_2$, $S_1 = P_1 \setminus P_2$, and $S_3 = P_2 \setminus P_1$.

Without constraints, n samples can be inferred using $\lceil \log(n) \rceil + 1$ pools (Theorem S1, also indirectly follows from Prabhu *et al.* (2009)). However, heterogeneous viral samples impose restrictions on pools composition: (i) the number of samples per pool should not be too high; (ii) samples with drastically different viral titers or samples, which may be epidemiologically related, should not be mixed together. These constraints make the pool design problem computationally hard. We formalize the constraints and formulate the pool design problem as an optimization problem on graphs.

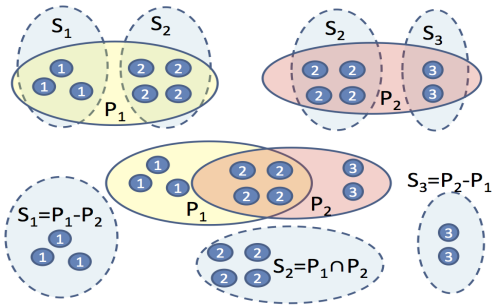

Fig. 2. 2 pools for 3 samples: S_1 has 3, S_2 has 4 and S_3 has 2 variants.

For a set of samples $\mathcal{S} = \{S_1, ..., S_n\}$ consider a *samples compatibility* graph $G = G(S)$ with the vertex set $V = V(G)$ and the edge set $E = E(G)$, such that $V(G) = S$ and $S_iS_j \in E(G)$ if and only if the samples S_i and S_j could be mixed together. As aforementioned, information on viral load, epidemiological linkage, geographic location, age/social groups, time of infection, risk factors, etc. may be used to determine compatibility of specimens. Every feasible pool is a clique of the graph G. Let T be an upper bound for pools size. The problem of optimal pool design for viral samples sequencing can be formulated as follows:

Viral Sample Pool Design (VSPD) Problem: *given a graph* $G = (V, E)$ *and* $T > 0$, *find a minimum set of cliques* $\mathcal{P} = \{P_1, ..., P_m\}$ *such that 1)* $\cup_{i=1}^{m} P_i = V$*; 2) for every* $u, v \in V$ *there is a clique* $P_i \in \mathcal{P}$ *separating* u *and* v*; 3)* $|P_i| \leq T$ *for every* $i = 1, ..., m$.

Due to the condition 2), at most one vertex $v \in V(G)$ is not covered by a clique from $\mathcal{P}$. Thus, any family of cliques satisfying 2) and 3) can be forced to satisfy 1) by adding the single clique $\{v\}$. Therefore the condition 1) is not essential and can be dropped.

VSPD is NP-hard (Theorem S2), and we propose a heuristic (Algorithm 1) for it. We consider a graph H with $V(H) = V$ and $ij \in E(H)$ if and only if the pair of vertices (i, j) is not separated yet. Initially, H is a complete graph. For $A \subseteq V$ a *cut* in the graph H is the pair $(A, V \setminus A)$, the *size of the cut* $c(A, V \setminus A)$ is the number of edges with one end in A and the other end in $V \setminus A$. At each iteration, Algorithm 1 finds and adds to the solution a locally optimal pool, i.e. a clique of G that separates the maximal number of non-separated samples (see example in Fig. S2).

Algorithm 1 Viral Sample Pool Design Algorithm

1: $\mathcal{P} \leftarrow \emptyset$; $H \leftarrow$ complete graph on n vertices
2: **while** $E(H) \neq \emptyset$ **do**
3: find a subset $A_x \subseteq V$ such that $|A_x| \leq T$, A_x is a clique of the graph G and $c(A_x, V \setminus A_x)$ in the graph H is maximal.
4: $\mathcal{P} \leftarrow \mathcal{P} \cup \{A_x\}$
5: remove from H all edges uv with $u \in A_x$ and $v \in V \setminus A_x$.
6: **end while**
7: **return** $\mathcal{P}$

The crucial step of Algorithm 1 is locally optimal pool finding (step 3), which represents a previously unstudied discrete optimization problem further referred as *Locally Optimal Pool (LOP) Problem*. LOP is not approximable within the factor $n^{1-\varepsilon}$ for any $\varepsilon > 0$ (Theorem S3), and it can be reformulated as follows: find a partition $X = (A_x, B_x)$ of the set V minimizing the function

$$f(X) = c(A_x, B_x) - M|E(\overline{G}[A_x])| \quad (1)$$

subject to the constraint $|A_x| \leq T$. Here $M = |E(H)| + 1$, $\overline{G}$ is a complement of a graph G, and $\overline{G}[A_x]$ is the subgraph of $\overline{G}$ induced by a set A_x. So, $f(X) \geq 0$ if and only if A_x is a clique. Therefore, for any optimal solution of the problem (1), the set A_x is a clique.

We propose a heuristic to solve the problem (1). Initially, we relax the constraint $|A_x| \leq T$. For a vertex $v \in A_x$ consider the solution $X' = (A_{x'}, B_{x'}) = (A_x \setminus \{v\}, B_x \cup \{v\})$. Then

$$\Delta_1 = f(X') - f(X) = deg^H_{A_x}(v) - deg^H_{B_x}(v) + M deg^{\overline{G}}_{A_x}, \quad (2)$$

where $deg^H_U(v)$ denotes the number of vertices from the set $U \subseteq V$ adjacent to v in a graph H. In particular, if v is non-adjacent to some vertex $u \in A_x$, then $\Delta_1 > 0$. For $v \in B_x$ and the solution $X' = (A_{x'}, B_{x'}) = (A_x \cup \{v\}, B_x \setminus \{v\})$ we have

$$\Delta_2 = f(X') - f(X) = deg^H_{B_x}(v) - deg^H_{A_x}(v) - M deg^{\overline{G}}_{A_x} \quad (3)$$

According to (2) and (3), any initial solution can be iteratively improved by moving vertices from one part of the partition to the other until a locally optimal solution (A_l, B_l) cannot be further improved. According to (2), A_l is a clique. However, the objective function value in a local optimum may be significantly lower than the value of the global optimum. It is also possible that $c(A_l, B_l) = 0$, when $E(H) \neq \emptyset$, which will cause Algorithm 1 to go into an infinite loop. To overcome these problems we use a *tabu search* strategy. The basic idea is that if after the moving of a vertex v the algorithm arrives at a local optimum, its value is compared with the current best solution (A^*, B^*), v is moved back and the moving of v is prohibited for the next k_t iterations. The process stops, when it reaches a local optimum, which has been visited before with the same configuration of the algorithm states. Finally, the set A^* is reduced to the allowed size. This idea is implemented in Algorithm 2 (see also S3 and Fig. S3). The default value of k_t is 1. If $c(A^*, B^*) = 0$, we increase k_t by one and repeat Algorithm 2.

2.2 Deconvolution of viral samples from pools

2.2.1 Deconvolution using generalized intersections and differences Let $\mathcal{P}$ be the set of pools designed using Algorithm 1 and sequenced by NGS. As aforementioned, the obtained reads theoretically can be assigned to samples using set-theoretic intersections and differences of pools. However, owing to the heterogeneity of viral populations and sampling bias, individual viral variants and even viral subpopulations sequenced from a certain sample may be different in each pool containing that sample (see Fig. S1). It hampers the usage of straightforward set-theoretic operations.

For a pool P_i, let $\mathcal{S}(P_i)$ be a set of IDs of samples mixed in it. In particular, we consider each individual sample R_j as a pool with $|S(R_j)| = 1$. A *generalized intersection* of pools P_1 and P_2 is a pool $P_1 \overline{\cap} P_2$ with $\mathcal{S}(P_1 \overline{\cap} P_2) = \mathcal{S}(P_1) \cap \mathcal{S}(P_2)$, consisting of sequences from $P_1 \cup P_2$ that belong to the samples from $\mathcal{S}(P_1) \cap \mathcal{S}(P_2)$. A *generalized difference* $P_1 \overline{\setminus} P_2$ is a pool with $\mathcal{S}(P_1 \overline{\setminus} P_2) = \mathcal{S}(P_1) \setminus \mathcal{S}(P_2)$ that contains sequences of the set $P_1 \setminus (P_1 \overline{\cap} P_2)$.

Algorithm 2 Locally Optimal Pool Problem Algorithm

Find the solution (X, Y) of Max-Cut problem using 0.5-approximation algorithm (Khuller *et al.* (2007)) applied to the graph H. Consequently apply the stages 2-28 to two initial solutions: $(A^0, B^0) = (X, Y)$ and $(A^0, B^0) = (Y, X)$
$i \leftarrow 0$; $\mathrm{tabu}^i \leftarrow (0, ..., 0)$; optStates $\leftarrow \emptyset$; moveList $\leftarrow \emptyset$; $(A^*, B^*) = (A^0, B^0)$.
while true do
  for every $u \in V$ calculate
$$\delta_u \leftarrow \begin{cases} \Delta_1, \text{ if } u \in A^i \text{ and } \mathrm{tabu}_u^i = 0 \text{ (see (2))}; \\ \Delta_2, \text{ if } u \in B^i \text{ and } \mathrm{tabu}_u^i = 0 \text{ (see (3))}; \\ 0, \text{ if } \mathrm{tabu}_u^i > 0. \end{cases}$$
  $\delta^* \leftarrow \max\limits_{u \in V}\{\delta_u\}$; $u^* \leftarrow \arg\max\limits_{u \in V}\{\delta_u\}$
  Update the tabu state:
$$\mathrm{tabu}_j^{i+1} \leftarrow \begin{cases} \mathrm{tabu}_j^i - 1, \text{ if } \mathrm{tabu}_j^i > 0; \\ 0, \text{ otherwise .} \end{cases}$$
  if $\delta^* > 0$ **then**
    update the current cut by moving the vertex u^*:
$$(A^{i+1}, B^{i+1}) \leftarrow \begin{cases} (A^i \setminus \{u^*\}, B^i \cup \{u^*\}), \text{ if } u^* \in A^i; \\ (A^i \cup \{u^*\}, B^i \setminus \{u^*\}), \text{ if } u^* \in B^i. \end{cases}$$
    Push u^* into the stack moveList; $i \leftarrow i + 1$
  else
    if $f(A^i, B^i) > f(A^*, B^*)$ **then**
      $(A^*, B^*) \leftarrow (A^i, B^i)$
    end if
    if moveList $\neq \emptyset$ **then**
      Pop a vertex v from the stack moveList, restore the cut, which was changed by the move of v and forbid to move v for the next k_t iterations: $\mathrm{tabu}_v^{i+1} \leftarrow k_t$;
$$(A^{i+1}, B^{i+1}) = \begin{cases} (A^i \setminus \{v\}, B^i \cup \{v\}), \text{ if } v \in A^i; \\ (A^i \cup \{v\}, B^i \setminus \{v\}), \text{ if } v \in B^i. \end{cases}$$
    end if
    $s \leftarrow ((A^{i+1}, B^{i+1}), \mathrm{tabu}^{i+1})$
    if $s \notin$ optStates **then**
      $i \leftarrow i+1$; optStates $\leftarrow$ optStates $\cup \{s\}$; **continue**
    else
      while $|A^*| > T$ **do**
        $a^* \leftarrow \arg\min_{a \in A^*}\{deg^H_{B^*}(a) - deg^H_{A^*}(a)\}$
        $(A^*, B^*) \leftarrow (A^* \setminus \{a\}, B^* \cup \{a\})$
      end while
      return A^*
    end if
  end if
end while

Individual samples are inferred from pools by a sequence of generalized intersections and differences (Algorithm 3). Generalized differences reduce to generalized intersections, which are calculated using a clustering-based approach (Algorithm 4). In some cases, when samples with substantial difference in heterogeneity are mixed together, highly heterogeneous samples can be partitioned into multiple clusters, while samples with lower heterogeneity are joined into one cluster. Such clustering may lead to incorrect detection of generalized intersections and consecutive fail of samples separation. To avoid this effect, the parameter W of Algorithm 4 with the default value $W = 2$ is introduced. If certain samples are not found by Algorithm 3 (i.e. some sets from $\mathcal{R}$ are empty), we increase W and repeat Algorithm 3.

Algorithm 3 Pools Deconvolution (PD) Algorithm

Input The set of pools $\mathcal{P} = \{P_1, ..., P_m\}$
Output The set of individual samples $\mathcal{R} = \{R_1, ..., R_n\}$
define two queues $\mathcal{Q} \leftarrow \mathcal{P}$ and $\mathcal{R}' \leftarrow \emptyset$.
while $\mathcal{Q} \neq \emptyset$ **do**
  $P' \leftarrow$ the first element of $\mathcal{Q}$; $\mathcal{Q} \leftarrow \mathcal{Q} \setminus P'$
  if $|S(P')| = 1$ **then**
    $\mathcal{R}' \leftarrow \mathcal{R}' \cup \{P'\}$
  end if
  find the first element $P'' \in \mathcal{Q}$ such that $S(P') \cap S(P'') \neq \emptyset$
  if such element exists **then**
    $\mathcal{Q} \leftarrow \mathcal{Q} \cup \{P' \overline{\cap} P''\}$
    if $S(P') \setminus S(P'') \neq \emptyset$ **then**
      $\mathcal{Q} \leftarrow \mathcal{Q} \cup \{P' \overline{\setminus} P''\}$
    end if
    if $S(P'') \setminus S(P') \neq \emptyset$ **then**
      $\mathcal{Q} \leftarrow \mathcal{Q} \cup \{P'' \overline{\setminus} P'\}$
    end if
  end if
end while
return last n elements of $\mathcal{R}'$

Algorithm 4 Generalized Intersection (GI) Algorithm

Input Pools P_1, P_2, parameter $W \geq 1$
Output The generalized intersection $P_1 \overline{\cap} P_2$
$K \leftarrow |\mathcal{S}(P_1) \cup \mathcal{S}(P_2)|$
Partition $P_1 \cup P_2$ into WK clusters using Maximum Likelihood k-Clustering described in Subsection 2.2.2 (Algorithm 5).
return union of clusters containing reads from both P_1 and P_2.

2.2.2 Maximum Likelihood k-Clustering of viral samples In this section we consider **Viral Sample Clustering (VSC) Problem:** *given a set $\mathcal{R}$ of NGS reads drawn from a mix of k' viral samples, partition $\mathcal{R}$ into $k = Wk'$ subsets consisting of reads from a single sample.* The presence of numerous variants, extreme heterogeneity of viral populations and sequencing errors make VSC challenging. Although a commonly used clustering objective is to minimize intra-cluster distances or distance to cluster centers (e.g., the k-means algorithm), we propose to use a statistically sound objective of maximizing likelihood. An input of our algorithm is a multiple sequence alignment of $\mathcal{R}$ represented as a matrix with columns corresponding to the consensus positions and rows corresponding to aligned reads. Our model assumes that each read is emitted by a particular genotype. The proposed clustering (a) finds k genotypes $G^1, \ldots, G^k$ that most likely emitted $\mathcal{R}$; (b) assigns each read to a cluster corresponding to a genotype that most likely emitted it.

Formally, given a set of reads C_i, a *genotype* $G^i = g(C^i)$ is a matrix with columns corresponding to an alignment positions and 5 rows corresponding to the alleles $\{\mathtt{a}, \mathtt{c}, \mathtt{t}, \mathtt{g}, \mathtt{d}\}$, where each entry $G^i_{e,m}$ is a frequency of allele e in m-th position among all variants

in C^i. Given a set of reads $\mathcal{R}$, an optimal *k-genotype* is a set $\mathcal{G}^* = \{G^1, \ldots, G^k\}$ of k distinct genotypes that most likely emitted $\mathcal{R}$:

$$G^* = arg \max_{|\mathcal{G}|=k} \prod_{r \in \mathcal{R}} Pr(r|\mathcal{G})^{o_r},$$

where $\Pr(r|\mathcal{G}) = \sum_{i=1}^{k} f_i \Pr(r|G^i)$ is the probability to observe read $r = (r_1, ..., r_m)$, o_r is its observed multiplicity, f_i is the frequency of the genotype G^i and

$$\Pr(r|G^i) = \prod_{m=1}^{L} G^i_{r_m,m}, \tag{4}$$

The log-likelihood of the set of genotypes $\mathcal{G}$ equals to $\ell(G) = \sum_{r \in \mathcal{R}} o_r \log \Pr(r|\mathcal{G})$. We iteratively estimate the missing data f_i and $p_{i,r}$ (the frequency of genotype G^i and the portions of reads originated from G^i, that mathches r) using Expectation Maximization algorithm and solve the easier optimization problem of maximizing the log-likelihood of the hidden model

$$\ell_{hid}(\mathcal{G}) = \sum_{r \in \mathcal{R}} \sum_{i=1}^{k} p_{i,r} \log(f_i \Pr(r|G^i)).$$

Our clustering method is described in Algorithm 5. The parameters ϵ there is the mutation rate.

Algorithm 5 Maximum Likelihood k-Clustering Algorithm

1: Find k *seed reads* $s^1, ..., s^k$ by iteratively selecting the most frequent read maximizing the minimum Hamming distance to the previously selected reads. Then define the initial k-genotype $\mathcal{G}^0 = \{G^1, ..., G^k\}$ as follows:
$$G^i_{e,m} \leftarrow \begin{cases} 1 - 4\varepsilon, \text{ if } s^i_m = e; \\ \varepsilon, \text{ otherwise .} \end{cases}$$
2: $h_{i,r} \leftarrow \Pr(r|G^i)$ (see (4)); $t \leftarrow 0$;
3: **repeat**
4: $f_i^{(0)} \leftarrow \frac{1}{k}$ for all $i = 1, \ldots k$; $\tau \leftarrow 0$;
5: **repeat**
6: For each read r and genotype $G^i \in \mathcal{G}^t$ compute $e_{i,r}$ - the expected number of reads emitted by G^i that match r:
$$p_{i,r} \leftarrow \frac{f_i^{(\tau)} \cdot h_{i,r}}{\sum_{j=1}^{k} f_j^{(\tau)} \cdot h_{j,r}}, \quad e_{i,r} \leftarrow o_r \cdot p_{i,r}$$
7: Calculate the updated frequency of each genotype $G^i \in \mathcal{G}^t$ as the portion of all reads emitted by G^i:
$$f_i^{(\tau+1)} \leftarrow \frac{\sum_{r \in \mathcal{R}} e_{i,r}}{\sum_{j=1}^{k} \sum_{r \in \mathcal{R}} e_{j,r}}, \; i = 1, ..., k; \; \tau \leftarrow \tau + 1;$$
8: **until** $\sum_{i=1}^{k} (f_i^{(\tau)} - f_i^{(\tau+1)})^2 \geq \delta$
9: Calculate the updated k-genotype $\mathcal{G}^{t+1} = \{G^1, ..., G^k\}$:
$$G^i_{e,m} \leftarrow \begin{cases} 1 - 4\varepsilon, \text{ if } e = \underset{e' \in \{\mathtt{a},\mathtt{c},\mathtt{t},\mathtt{g},\mathtt{d}\}}{\arg\max} \sum_{r \in \mathcal{R}: r_m = e'} p_{i,r}; \\ \varepsilon, \text{ otherwise .} \end{cases}$$
10: $h_{i,r} \leftarrow \Pr(r|G^i)$ (see (4)); $t \leftarrow t + 1$;
11: **until** $\mathcal{G}^t \neq \mathcal{G}^{t+1}$
12: Assign each read r to the cluster j_r, where $j_r \leftarrow arg \max_i p_{i,r}$

2.3 NGS errors and sequencing failures processing

Before applying deconvolution algorithms, the data are preprocessed to remove sequencing errors and PCR chimeras. Since errors may be sample-specific, the following pipeline is used: (1) each pool is partitioned into clusters using Algorithm 5; (2) NGS error correction algorithm is applied to each cluster. This algorithm is specific to sequencing platform and sequenced species. (3) Corrected reads from each cluster are used to reinstate pools.

Failure to recover sequences from some samples within certain pools may result in algorithm's inability to separate these samples. Given the aforementioned pools design constraints, use of appropriate PCR conditions and high NGS throughput, we expect that the probability of a complete loss of a sample within a pool is very low. Nevertheless, if sequencing of some samples within certain pools fails, the workflow summarized in Algorithm 6 allows to detect and eliminate the negative effects of the failures.

Algorithm 6 Failure Detection and Processing

Input Datasets $\mathcal{R} = \{R_1, ..., R_n\}$ produced by Algorithm 3.
1: Identify failed samples $\mathcal{S}_f \leftarrow \{j : R_j = \emptyset\}$ (the sample can be also considered failed, if it does not have a required number of reads, i.e. $|R_j| \leq \Delta$) and re-sequence them (either individually or using the same pooling framework). For every $j \in \mathcal{S}_f$ replace R_j by the obtained data set.
2: **for** every $i \in \{1, ..., n\} \setminus \mathcal{S}_f$ and $j \in \mathcal{S}_f$ **do**
3: $Q_{i,j} = R_i \overline{\cap} R_j$, $R_i \leftarrow R_i \setminus Q_{i,j}$, $R_j \leftarrow R_j \cup Q_{i,j}$
4: **end for**
5: **return** $\mathcal{R}$

2.4 Experimental pools and sequencing

Serum specimens collected from HCV-positive cases (Holodniy *et al.* (2012)) were used to sequence HCV HVR1 region. Seven samples $S_1, ..., S_7$ were mixed to form 4 pools $P_1, ..., P_4$ as follows: P_1 was created by mixing samples S_1, S_2, S_3, P_2 - S_4, S_5, S_6, S_7, P_3 - S_1, S_4, S_5 and P_4 - S_2, S_4, S_6. Specimens and pools were sequenced using 454 GS Junior System (454 Life Sciences, Branford, CT). Total nucleic acids extraction was performed using MagNA Pure LC Total Nucleic Acid Isolation Kit (Roche Diagnostics, Mannheim, Germany) and reverse-transcribed using SuperScript Vilo cDNA synthesis kit (Invitrogen, Carlsbad, CA).

HVR1 amplification was accomplished using two rounds of PCR. For the 1st round, regular region-specific primers were used. Forward and reverse tag sequences consisting of primer adaptors and multiple identifiers (MID) were added to the HVR1-specific nested primers. Pools were processed as a single specimen, tagged with a single MID. PCR products were pooled and amplified by emulsion PCR using the GS FLX Titanium Series Amplicon kit, and bi-directionally sequenced. The NGS reads were identified and separated using sample-specific MID tag identifiers. Low quality reads were removed using GS Run Processor v2.3 (Roche, 2010) and the obtained datasets were processed using error correction pipeline with algorithms KEC and ET (Skums *et al.* (2012a)).

3 RESULTS

3.1 Pools design

Pool design algorithm was evaluated using 3 sets of simulated data.

1) Complete graphs with $n = 4, ..., 1024$ vertices and with $T = \infty$. For every test instance, exactly $\lceil \log(n) \rceil + 1$ pools were constructed, coinciding with the theoretically justified estimation. Hence, in this case Algorithm 1 produces optimal solutions.

2) Random graphs, where each vertex v receives a random titer $w_v \in \{1, L\}$, and two vertices u and v are adjacent, when $|w_u - w_v| \leq R$. This family of instances represents *titer compatibility model*, i.e. it simulates the case when two samples could be mixed together only if their viral titers are not sufficiently different. 25000 test instances were generated with $n = 10, ..., 1000$, $L = 20$, $R = 4$ and with the pools sizes thresholds $T = n$, $55, 35, 25, 15$. For each n the mean reduction coefficient (the ratio of a number of pools and a number of samples) was calculated (Fig. 3, (a)). For $n = 1000$ the reduction of the number of sequencing runs varies from more than 21 for $T = n$ to 6 for $T = 15$.

3) Random graphs, where each edge is chosen with probability $p = 0.25, 0.5, 0.75, 1$ and pools sizes are bounded by $T = 35$. 20000 test instances with $n = 10, ..., 1000$ were generated (Fig. 3, (b)). A reduction of the sequencing runs number is also high, although it is generally lower than in 2) (from $\sim$ 13-fold reduction for $p = 1$ to more than 3-fold reduction for $p = 0.25$, $n = 1000$).

The reduction coefficient in all these cases is a decreasing function of n, which suggests a higher reduction for the larger n. Working time of the pool design algorithm is shown in Fig. S4.

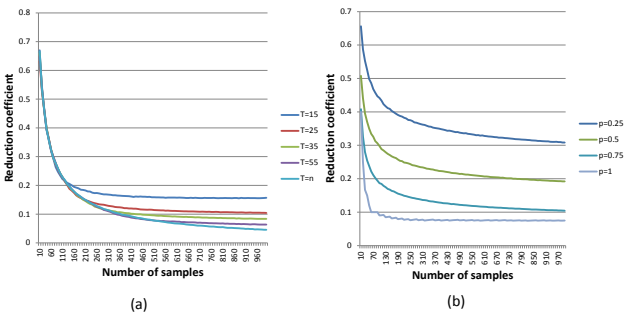

Fig. 3. Reduction coefficients for the pools generated by Algorithm 1 for (a) random titer compatibility model graphs; (b) random graphs.

3.2 Pools deconvolution

3.2.1 Simulated pools of experimental data. 450 test instances with $n = 10, ..., 150$ samples and with pool sizes thresholds $T = 15, 25, 35$ were generated using 155 HCV HVR1 samples from the collection of Centers for Disease Control (Dimitrova *et al.* (2012); Campo *et al.* (2014a); Lara *et al.* (2012); Campo *et al.* (2014b)). Samples were cleaned using KEC and ET (Skums *et al.* (2012a)). Test instances were generated as follows: (1) n samples were chosen randomly; (2) a random samples compatibility graph on n vertices was generated based on the titer compatibility model and pools were designed using Algorithm 1; (3) pools were created by taking $D = 10000$ randomly selected reads from the samples composing each pool (in order to simulate a sampling bias). The number of reads per pool corresponds to the sequencing settings, under which the data used for simulation were obtained (454 GS Junior System (454 Life Sciences, Branford, CT) with 8-10 MIDs per sequencing run).

For all test instances all samples were inferred, i.e. all n data sets produced by Algorithm 3 were non-empty. The number of reads, which were not classified into samples was extremely low (Fig. 4, (a)): in avearge 99.996% of reads ($T = 15$), 99.993% ($T = 25$) and 99.984% ($T = 35$) were assigned. An overwhelming majority of reads was classified correctly (Fig. 4, (b)): in average, 99.998% for $T = 15$, 99.982% for $T = 25$ and 99.959% for $T = 35$. There is no clear correlation between percentages of classified and correctly classified reads and the samples number.

We call an incorrect assignment of reads to samples *in silico contamination*. The average percentage of samples without in silico contamination ranges from 100% to 98.13% ($T = 15$), from 100% to 96.13% ($T = 25$) and from 100% to 93.8% ($T = 35$) (Fig. 4, (c)). In silico contaminants within contaminated samples in average constitute 0.163% ($T = 15$), 0.545% ($T = 25$) and 0.892% ($T = 35$) of all reads (Fig. 4, (d)). Root Mean Squeare Error (RMSE) of deconvoluted haplotypes frequencies estimation is in average 0.031%-0.107% ($T = 15$), 0.025%-0.139% ($T = 25$) and 0.028%-0.174% ($T = 35$) (Fig. 4, (e)). Both the percentages of in silico contaminated samples and RMSE increase with n.

The accuracy of samples deconvolution is affected by the number of allowed samples per pool. The algorithm is more accurate for smaller pools, although the accuracy for larger pools remains high. Working times of pools deconvolution are shown in Fig. S4.

To test failure detection and processing workflow, 150 test instances with $T = 15$ were generated, where in addition to steps (1)-(3) a random subset of up to 50% of pools was selected, and in each of these pools all sequences originated from a random subset of up to 25% of samples were removed. The instances were processed by Algorithms 3 and 6 (resequencing was simulated by taking sequences from the corresponding individual samples). For all test instances all samples were inferred, and the quality of deconvolution was comparable with the quality without failures (Fig. S5).

3.2.2 Experimental pools Experimental pools (Section 2.4) were deconvoluted using Algorithm 3, and the obtained samples (further refered as *p-samples*) were verified by comparison with the individually sequenced samples (*i-samples*). 10 references were taken from each i-sample, and the correctness of deconvolution was assessed by finding the closest reference to each p-sample sequence. Sequences were aligned using Muscle (Edgar (2004)).

In average, 259 unique haplotypes per p-sample were obtained, which exceeds the numbers obtained in other studies after the standard sequencing using 454 Junior System and error correction (Bull *et al.* (2011); Caraballo Cortes *et al.* (2013); Gregori *et al.* (2013)). 99.96% (5463 of 5465) of reads were correctly assigned to samples. Two reads assigned to sample S_7 showed a higher similarity to a reference from S_6. The subsequent analysis showed that these reads are distant from S_6 and S_7 as well as from each other: minimum distance from these reads to haplotypes from S_6 and S_7 is 25 and 26, respectively, and the distance between them is 20, while the mean distance among haplotypes of i-samples S_6 and S_7 is 3.64 bp (std 1.21 bp) and 6.12 bp (std 5.25 bp), respectively. Therefore, these 2 reads are likely to be sequencing artifacts.

In general, the percentage of haplotypes from i-samples found in p-samples was not high (Fig. 5, (a)), with an average of 14.66%. However, when the frequencies of these haplotypes were considered, the level of agreement was much higher, with an average total frequency of 56.94% (Fig. 5, (b)). In particular, all individually

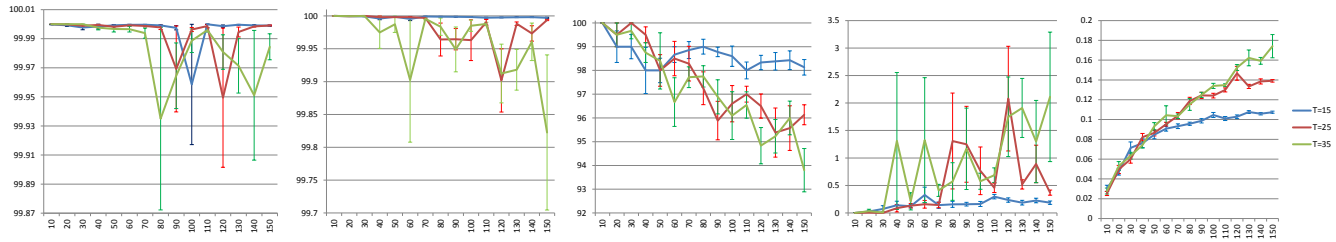

Fig. 4. (a) Percentage of classified reads (b) Percentage of correctly classified reads (c) Percentage of in silico contamination - free samples (d) Total frequency of in silico contaminants within contaminated samples (e) Root Mean Square Error of haplotypes frequencies estimation. Bars represent a standard error.

Table 1. Comparison of frequency distributions for i- and p-samples

	JSD	Corr (p-value)		JSD	Corr (p-value)
S_1	0.15	0.95 ($1.71\dot{1}0^{-77}$)	S_5	0.50	0.25 ($8.61\dot{0}^{-7}$)
S_2	0.57	0.30 (0.0023)	S_6	0.17	0.99 (0)
S_3	0.32	0.89 ($2.71\dot{1}0^{-173}$)	S_7	0.65	-0.07 (0.16)
S_4	0.37	0.66 ($4.14\dot{1}0^{-99}$)			

sequenced haplotypes with frequencies $\geq$ 10% and 72.73% of haplotypes with frequencies $\geq$ 5% were found in p-samples.

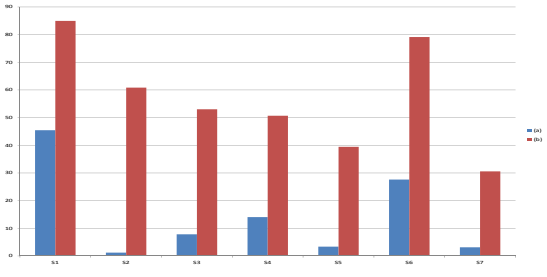

Fig. 5. (a) Percentage of haplotypes from i-samples found by pooling. (b) Total frequency of haplotypes from i-samples found by pooling

In general haplotypes from i- and p-samples cover the same areas of the sequence space, although some branches are formed by variants sequenced in only one of experiments (Fig. 6). The differences between haplotype frequencies distributions for i- and p-samples were measured using Jensen-Shannon Divergence (JSD) and correlation coefficient (Corr) (Table 1). JSD varies from 0.15% (S_1) to 0.65% (S_7). A correlation between frequencies distributions is positive and statistically significant for all samples except S_7, in which a large cluster of variants was not detected by the individual sequencing, but was found in the pooling experiment (Fig. 6).

4 DISCUSSION

In this study, we present a novel computational framework for massive NGS of highly mutable viruses. To the best of our

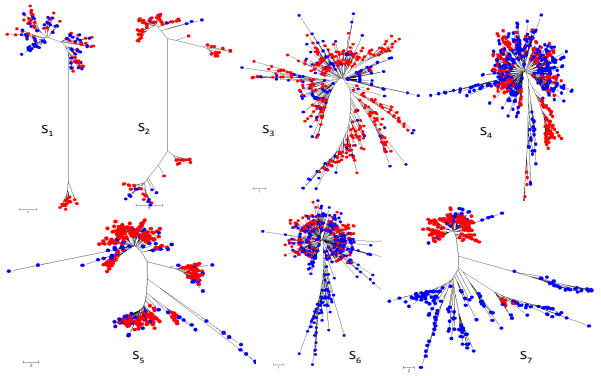

Fig. 6. Phylogenetic trees of viral populations of samples S_1-S_7. Haplotypes of i-samples and p-samples are shown in red and blue

knowledge, this is the first pooling framework applicable for sequencing of viral quasispecies, which takes into account extensive heterogeneity of viral populations, the large number of distinct viral variants sequenced from each sample and the effects of PCR and sampling biases. The proposed strategy drastically reduces the cost of sequencing per specimen (section S5), while still providing sufficient amount of information in support of molecular surveillance and other applications of viral sequences in clinical and epidemiological settings. The framework is applicable to viral agents infecting humans and animals and, with further development of experimental protocols, it should serve as a cost-effective foundation for molecular surveillance of infectious diseases.

Ultra-deep sequencing of viral samples produces a wide range of intra-host viral variants and allows for detecting even minor viral subpopulations. Pooling of several specimens reduces the depth of sequencing for each specimen, but this reduction is not detrimental, since each specimen is usually used in more than one pool. As specimen is tested more than once, the number of sequenced variants is increased, so representative sampling of viral subpopulations infecting each patient can be improved. The experiments conducted here showed that comparable subpopulations were recovered from individual specimens and from pools, at least at the pooling scale used in this study. Both individual sequencing and pooling produce sequences covering approximately

the same areas of the sequence space, thus providing a consistent structure of a viral population.

Repeat sampling from the same complex viral population often results in poorly matched sets of sequences, thus presenting a significant challenge to pools deconvolution. Such stochastic sampling has a potential to diminish the effectiveness of pool-sequencing and usefulness of the obtained sequences by impeding the correct allocation of sequences to samples, leaving some samples without sequences assigned or allocating only a fraction of the sequences. The clustering approach developed in this study significantly improves assignemt of sequences to samples and, thus, not only substantially overcomes the aforementioned potential pitfalls, but converts stochastic sampling into an advantage. Clustering also eliminates the detrimental effect of NGS errors, since erroneous reads tend to concentrate around correct haplotypes.

The sequencing cost and accuracy of deconvolution are two major measures of quality of our framework. These two measures are in conflict with each other: while increase in pool size improves cost-effectiveness of sequencing by reducing the number of sequencing runs, it reduces accuracy of deconvolution. Considering that deconvolution accuracy significantly depends on the genetic complexity of intra-host viral populations, an optimal pool size should be carefully selected for each virus and genomic region.

In conclusion, success of the pool-based sequencing of viral populations depends to a significant degree on the efficacy of sequence assignments and the risk of under-representation of viral variants from some samples, owing to PCR and sample biases. The pool design and clustering algorithms presented here substantially minimize the detrimental effect of these biases on the sequencing quality. Further reduction of the biases using generalizations of error-correcting codes and optimization of experimental conditions may further improve the strategy, facilitating its application to molecular surveillance and study of infectious diseases.

ACKNOWLEDGEMENT

We thank reviewers for helpful comments and Seth Sims and Vernard Martin (CDC/NCEZID) for help with CDC cluster.

REFERENCES

Alon, S. *et al.* (2011). Barcoding bias in high-throughput multiplex sequencing of mirna. *Genome Research*, **21**(9), 1506–1511.

Bansal, V. (2010). A statistical method for detection of variants from next-generation resequencing of dna pools. *Bioinformatics*, **26**(12), i318–i324.

Berman, P. *et al.* (2004). Tight approximability results for test set problems in bioinformatics. *Lecture Notes in Computer Science*, **3111**, 39–50.

Bull, R. *et al.* (2011). Sequential bottlenecks drive viral evolution in early acute hepatitis c virus infection. *PLoS Pathogens*, **7**(9).

Campo, D. *et al.* (2014a). Drug-resistance of a viral population and its individual intra-host variants during the first 48 hours of therapy. *Clinical Pharmacology and Therapeutics*, **95**(6), 627–635.

Campo, D. *et al.* (2014b). Next-generation sequencing reveals large connected networks of intra-host hcv variants. *BMC Genomics*, **15**(Suppl 5: S4).

Caraballo Cortes, K. *et al.* (2013). Ultradeep pyrosequencing of hepatitis c virus hypervariable region 1 in quasispecies analysis. *BioMed Research International*.

Culasso, A. *et al.* (2014). Intra- host evolution of multiple genotypes of hepatitis c virus in a chronically infected patient with hiv along a 13-year follow-up period. *Virology*, **449**, 317–327.

Deakin, C. T. *et al.* (2014). Impact of next-generation sequencing error on analysis of barcoded plasmid libraries of known complexity and sequence. *Nucleic Acids Research*.

Dierynck, I. *et al.* (2014). Deep sequencing analysis of the hcv ns3-4a region confirms low prevalence of telaprevir-resistant variants at baseline and end of the realize study. *J Infect Dis*.

Dimitrova, Z. *et al.* (2012). Assessments of intra- and inter-host diversity of hepatitis c virus using next generation sequencing and mass spectrometry. *In Silico Biol.*, **11**(5), 183–192.

Du, D.-Z. *et al.* (2006). *Pooling Design and Nonadaptive Group Testing: Important Tools for DNA Sequencing*, volume 18. World Scientific Publishing Company. Series on Applied Mathematics.

Duma, D. *et al.* (2013). Accurate decoding of pooled sequenced data using compressed sensing. *Lecture Notes in Computer Science*, **8126**, 70–84.

Edgar, R. (2004). Muscle: multiple sequence alignment with high accuracy and high throughput. *Nucleic Acids Res*, **32**(5), 1792–1797.

Erlich, Y. *et al.* (2009). Dna sudoku - harnessing high-throughput sequencing for multiplexing specimen analysis. *Genome Res*, **19**, 1243–1253.

Golan, D. *et al.* (2012). Weighted pooling-practical and cost-effective techniques for pooled high-throughput sequencing. *Bioinformatics*, **28**(12), i197–i206.

Gregori, J. *et al.* (2013). Ultra-deep pyrosequencing (udps) data treatment to study amplicon hcv minor variants. *PLoS ONE*, **8**(12).

He, D. *et al.* (2011). Genotyping common and rare variation using overlapping pool sequencing. *BMC Bioinformatics*, **12**(Suppl 6).

Holodniy, M. *et al.* (2012). Results from a large-scale epidemiologic look - back investigation of improperly reprocessed endoscopy equipment. *Infect Control Hosp Epidemiol*, **33**(7), 649–656.

Khuller, S. *et al.* (2007). *Greedy methods*, chapter Handbook of Approximation Algorithms and Metaheuristics. Chapman and Hall/CRC.

Lara, J. *et al.* (2011-2012). Coordinated evolution among hepatitis c virus genomic sites is coupled to host factors and resistance to interferon. *In Silico Biol.*, **11**(5-6), 213–224.

Lonardi, S. *et al.* (2013). Combinatorial pooling enables selective sequencing of the barley gene space. *PLoS Comput Biol*, **9**(4).

Metzner, K. J. *et al.* (2009). Minority quasispecies of drug-resistant hiv-1 that lead to early therapy failure in treatment-naive and -adherent patients. *Clin Infect Dis*, **48**(2), 239–247.

Palmer, B. A. *et al.* (2012). Insertion and recombination events at hypervariable region 1 over 9.6 years of hepatitis c virus chronic infection. *J Gen Virol*, **93**, 2614–2624.

Prabhu, S. *et al.* (2009). Overlapping pools for high-throughput targeted resequencing. *Genome Res*, **19**(7), 1254–1261.

Ramachandran, S. *et al.* (2011). Temporal variations in the hepatitis c virus intrahost population during chronic infection. *J Virol*, **85**(13), 6369–6380.

Shental, N. *et al.* (2010). Identification of rare alleles and their carriers using compressed se(que)nsing. *Nucleic Acids Res*, **38**, 1–22.

Skums, P. *et al.* (2012a). Efficient error correction for next-generation sequencing of viral amplicons. *BMC Bioinformatics*, **13**(Suppl 10:S6).

Skums, P. *et al.* (2012b). Numerical detection, measuring and analysis of differential interferon resistance for individual hcv intra-host variants and its influence on the therapy response. *In Silico Biol*, **11**(5), 263–269.

Wang, W. *et al.* (2014). High- resolution quantification of hepatitis c virus genome-wide mutation load and its correlation with the outcome of peginterferon-alpha2a and ribavirin combination therapy. *PLoS ONE*, **9**(6).

Wertheim, J. *et al.* (2014). The global transmission network of hiv-1. *J Infect Dis*, **209**(2), 304–313.

Wu, W. *et al.* (2006). On error-tolerant dna screening. *Discrete Applied Mathematics*, **154**, 1753–1758.